

Computationally-Efficient Visual Inertial Odometry for Autonomous Vehicle

by

©Trung Nguyen

A Thesis submitted to the School of Graduate Studies

in partial fulfillment of the requirements for the degree of

Ph.D.

Faculty of Engineering and Applied Science

Memorial University of Newfoundland

May 2019

St. John's

Newfoundland

Abstract

This thesis presents the design, implementation, and validation of a novel nonlinear-filtering based Visual Inertial Odometry (**VIO**) framework for robotic navigation in GPS-denied environments. The system attempts to track the vehicle's ego-motion at each time instant while capturing the benefits of both the camera information and the Inertial Measurement Unit (**IMU**). VIO demands considerable computational resources and processing time, and this makes the hardware implementation quite challenging for micro- and nano-robotic systems. In many cases, the VIO process selects a small subset of tracked features to reduce the computational cost. VIO estimation also suffers from the inevitable accumulation of error. This limitation makes the estimation gradually diverge and even fail to track the vehicle trajectory over long-term operation. Deploying optimization for the entire trajectory helps to minimize the accumulative errors, but increases the computational cost significantly. The VIO hardware implementation can utilize a more powerful processor and specialized hardware computing platforms, such as Field Programmable Gate Arrays, Graphics Processing Units and Application-Specific Integrated Circuits, to accelerate the execution. However, the computation still needs to perform identical computational steps with similar complexity. Processing data at a higher frequency increases energy consumption significantly. The development of advanced hardware systems is also expensive and time-consuming. Consequently, the approach of developing an efficient algorithm will be beneficial with or without hardware acceleration. The research described in this thesis proposes multiple solutions to accelerate the visual inertial odometry computation while maintaining a comparative estimation accuracy over long-term operation among state-of-the-art algorithms.

This research has resulted in three significant contributions. First, this research involved the design and validation of a novel nonlinear filtering sensor-fusion algorithm using trifocal tensor geometry and a cubature Kalman filter. The combination has handled the system nonlinearity effectively, while reducing the computational cost and system complexity significantly. Second, this research develops two solutions to address the error accumulation issue. For standalone self-localization projects, the first solution applies a local optimization procedure for the measurement update, which performs multiple corrections on a single measurement to optimize the latest filter state and covariance. For larger navigation projects, the second solution integrates VIO with additional pseudo-ranging measurements between the vehicle and multiple beacons in order to bound the accumulative errors. Third, this research develops a novel parallel-processing VIO algorithm to speed up the execution using a multi-core CPU. This allows the distribution of the filtering computation on each core to process and optimize each feature measurement update independently.

The performance of the proposed visual inertial odometry framework is evaluated using publicly-available self-localization datasets, for comparison with some other open-source algorithms. The results illustrate that a proposed VIO framework is able to improve the VIO's computational efficiency without the installation of specialized hardware computing platforms and advanced software libraries.

To my Mother, Father and Brother

Acknowledgements

It is an exclusive privilege of being advised and working under the supervision of Dr. George K. I. Mann, Dr. Andrew Vardy and Dr. Raymond G. Gosine for my Ph.D. degree. Their invaluable lessons, academic guidance as well as encouragement improve my professional skills significantly. I would like to express my sincere and most profound appreciation to them and their values they give me to finish my thesis.

To thesis examiners, Intelligent Systems Laboratory (ISLab), Autonomous Ocean Systems Laboratory (AOSL), the faculty and staff of engineering, I would like to thank them for their assistance, patience and inspirations.

This research is supported by Research & Development Corporation of Newfoundland and Labrador Ocean Industries Student Research Awards (5474-1774-101), Natural Sciences and Engineering Research Council of Canada (NSERC), School of Graduate Studies, Memorial University of Newfoundland and C-CORE J. I. Clark Chair.

Last but not least, I would like to express my sincere gratitude to my parents Tam Nguyen and Van Bui, and my brother Long Nguyen for their love, support, and encouragement throughout my life.

Contents

| | |
|--|------------|
| Abstract | ii |
| Acknowledgements | v |
| List of Tables | x |
| List of Figures | xi |
| Abbreviations | xvi |
| 1 Introduction | 1 |
| 1.1 Motivation - Autonomous Vehicles | 1 |
| 1.2 Autonomous Navigation in GPS-denied Environment | 3 |
| 1.2.1 Visual Inertial Navigation System | 3 |
| 1.2.2 Issues Related to Visual Inertial Odometry Filtering | 6 |
| 1.3 Thesis Problem Statement | 8 |
| 1.3.1 Problem I: Computational Cost of Visual Measurement Model . . . | 8 |
| 1.3.2 Problem II: Nonlinear Filtering for Visual Inertial Odometry . . . | 9 |
| 1.3.3 Problem III: Error Accumulation over Long-term Operation . . . | 10 |

| | | |
|----------|--|-----------|
| 1.3.4 | Problem IV: Parallel Processing to Accelerate the Computation . . . | 11 |
| 1.4 | Objectives and Expected Contributions | 11 |
| 1.5 | Organization of the Thesis | 13 |
| 2 | Background and Literature Review | 16 |
| 2.1 | Overview of Visual Inertial Odometry | 16 |
| 2.1.1 | General System Architecture Design | 17 |
| 2.1.2 | Classification of Tightly-coupled VIO Algorithms | 21 |
| 2.2 | Filtering Methods for VIO Algorithm | 23 |
| 2.3 | VIO Hardware Implementation | 27 |
| 2.3.1 | Group B: Developing a Computationally-efficient VIO Algorithm . | 27 |
| 2.3.2 | Group C: Accelerating Execution Using Specialized Hardware . . . | 28 |
| 2.4 | VIO Augmentation | 30 |
| 2.5 | Summary | 31 |
| 3 | Developing Computationally-Efficient Nonlinear Cubature Kalman Filtering for Visual Inertial Odometry | 32 |
| 3.1 | Introduction | 33 |
| 3.2 | System Description | 35 |
| 3.2.1 | System Coordinates and Essential Notation | 35 |
| 3.2.2 | IMU Sensor Model | 37 |
| 3.3 | IMU Filter State Propagation | 38 |
| 3.3.1 | Filter State Formation | 38 |
| 3.3.2 | Filter Propagation | 39 |
| 3.3.3 | Filter Measurement Update with Cubature Rule | 41 |

| | | |
|----------|--|-----------|
| 3.4 | Trifocal Tensor Geometry Based Measurement Model | 45 |
| 3.4.1 | Point Transfer Using Trifocal Tensor Geometry | 46 |
| 3.4.2 | Comparison between TTG-based Approach and 3D Feature-point Approach | 50 |
| 3.5 | Experimental Results | 53 |
| 3.5.1 | Experiments with KITTI Dataset | 53 |
| 3.5.2 | Experiments with EuRoC Dataset | 60 |
| 3.5.3 | Experiments with Handheld VIO Device | 62 |
| 3.6 | Cubature Information Filtering | 66 |
| 3.6.1 | Transformation to Information Domain | 66 |
| 3.6.2 | Experiments for Cubature Information Filter | 69 |
| 3.7 | Conclusions | 74 |
| 4 | Accurate Visual Inertial Odometry for Long-Term Trajectory Operations With- out Using a Map | 75 |
| 4.1 | Introduction | 76 |
| 4.2 | CKF-based VIO's Error-Accumulation Issue | 78 |
| 4.3 | Solution 1: Iterated Cubature Kalman Filter | 81 |
| 4.3.1 | Maximum Likelihood Estimate Based Optimization | 81 |
| 4.3.2 | The System Architecture of Iterated CKF | 84 |
| 4.3.3 | Experimental Validation of Solution 1 | 86 |
| 4.4 | Solution 2: Pseudo-range Measurements | 91 |
| 4.4.1 | Sequential-Sensor-Update Approach | 91 |
| 4.4.2 | Experimental Validation of Solution 2 | 95 |

| | | |
|----------|--|------------|
| 4.5 | Discussion and Conclusions | 95 |
| 5 | Efficient Parallel Processing Solution to Accelerate Visual Inertial Odometry | |
| | Execution on a Generic Computer | 99 |
| 5.1 | Introduction | 100 |
| 5.2 | Preliminaries | 104 |
| 5.2.1 | Matrix Lie Groups | 104 |
| 5.2.2 | Problem Formulation | 105 |
| 5.3 | Filter State Formation and Prediction | 108 |
| 5.4 | Filter Measurement Update | 109 |
| 5.5 | System Architecture | 112 |
| 5.5.1 | Distribute Computation on Multiple Cores | 112 |
| 5.5.2 | Optimization during Measurement Update | 114 |
| 5.6 | Experimental Validation | 115 |
| 5.7 | Conclusions | 128 |
| 6 | Conclusion and Outlook | 130 |
| 6.1 | Research Summary Based on Objective I | 131 |
| 6.2 | Research Summary Based on Objective II | 132 |
| 6.3 | Research Summary Based on Objective III | 132 |
| 6.4 | Summary of Contributions | 133 |
| 6.5 | Future Research Directions | 136 |
| | BIBLIOGRAPHY | 138 |

List of Tables

| | | |
|-----|--|-----|
| 1.1 | Notable open-source VIO (regular type) and SLAM-based (<i>italic</i> type) developments in recent years | 5 |
| 2.1 | Overview of publicly-available datasets | 20 |
| 2.2 | The most relevant VIO papers considered in this thesis | 22 |
| 3.1 | Experimental results for KITTI datasets with different travel distance [m] . | 58 |
| 5.1 | Technical specifications of hardware platforms | 117 |

List of Figures

| | | |
|-----|--|----|
| 1.1 | Some popular MAV's flight time for manual operation | 2 |
| 1.2 | Visual SLAM application | 4 |
| 2.1 | General tightly-coupled VIO system architecture | 17 |
| 2.2 | Illustration of the data updating in VIO system | 23 |
| 3.1 | The VIO coordinate system | 36 |
| 3.2 | Illustration of the spherical-radical transformation using in CKF [18] | 42 |
| 3.3 | System architecture of the proposed algorithm | 45 |
| 3.4 | Illustration of the 3D feature-point reconstruction | 46 |
| 3.5 | Trifocal tensor incidence relation (point-line-point) for three views I_1 , I_2 , and I_3 with camera viewpoints O_1 , O_2 , and O_3 | 47 |
| 3.6 | Illustration of the TTG based approach. The red line is the point transfer using TTG while the blue line is the feature-tracking pipeline | 48 |

| | | |
|------|---|----|
| 3.7 | Example of point transfer using TTG. Images are obtained from KITTI dataset [56]. The point-line-point relation, which is represented with different colours, is established between the feature \mathbf{m}_1 in I_1 , the line \mathbf{l}_2 (thicker line) in I_2 and the feature \mathbf{m}_3 in I_3 . The feature matching is presented with the thinner line connecting three features of three images. | 49 |
| 3.8 | Structure of TTG-based approach and 3D feature-point approach | 51 |
| 3.9 | The results of the predicted measurements when verifying two approaches with KITTI dataset [56] | 51 |
| 3.10 | Average processing time to predict one feature in each approach | 52 |
| 3.11 | Some camera images in KITTI dataset [56] | 54 |
| 3.12 | CKF estimation presented in Google map: (a) 2011_09_26_0095, (b) 2011_09_30_0020, (c) 2011_09_30_0033, (d) 2011_09_30_0034 | 55 |
| 3.13 | CKF estimation of velocity in dataset 2011_09_30_0034 | 56 |
| 3.14 | CKF estimation of accelerometer bias in dataset 2011_09_30_0034 | 56 |
| 3.15 | CKF estimation of gyroscope bias in dataset 2011_09_30_0034 | 56 |
| 3.16 | Processing time evaluation of CKF and EKF | 59 |
| 3.17 | Experiments with EuRoC dataset [26] | 61 |
| 3.18 | A handheld VIO device | 63 |
| 3.19 | Snapshots during the experiment with the handheld VIO device | 63 |
| 3.20 | The trajectory of the handheld VIO device is estimated using the proposed algorithm and compared with the VINS-Mono estimate. The blue line is the map of the lab EN1037 generated by a laser scanner | 64 |
| 3.21 | Positional and rotational estimations of the proposed algorithm and VINS-Mono | 65 |

| | | |
|------|---|----|
| 3.22 | CIF system architecture with multiple measurement updates | 68 |
| 3.23 | Experimental results drawn on Google map using dataset 2011_09_30_0034 | 70 |
| 3.24 | RMSE of positional estimation with KITTI dataset 2011_09_30_0034 | 70 |
| 3.25 | Rotation error evaluation of KITTI dataset 2011_09_30_0034 | 70 |
| 3.26 | Experimental results drawn on Google map using dataset 2011_09_30_0020 | 71 |
| 3.27 | RMSE of positional estimation with KITTI dataset 2011_09_30_0020 | 71 |
| 3.28 | Rotation error evaluation of KITTI dataset 2011_09_30_0020 | 71 |
| 3.29 | Experimental results drawn on Google map using dataset 2011_09_30_0033 | 72 |
| 3.30 | RMSE of positional estimation with KITTI dataset 2011_09_30_0033 | 72 |
| 3.31 | Rotation error evaluation of KITTI dataset 2011_09_30_0033 | 72 |
| 3.32 | Processing time evaluation of CKF and CIF | 73 |
| 4.1 | Position estimation in dataset 2011_09_30_0034 | 79 |
| 4.2 | Position estimation in 2011_09_30_0033 | 80 |
| 4.3 | The system architecture developed with iterated CKF | 84 |
| 4.4 | Experimental results presented in Google map for 2011_09_26_0087 | 86 |
| 4.5 | RMSE evaluation in the experiment with dataset 2011_09_26_0087 | 87 |
| 4.6 | Rotation errors in the experiment with dataset 2011_09_26_0087 | 87 |
| 4.7 | Experimental results presented in Google map for 2011_09_30_0034 | 88 |
| 4.8 | RMSE in the experiment with dataset 2011_09_30_0034 | 88 |
| 4.9 | Rotation errors in the experiment with dataset 2011_09_30_0034 | 88 |
| 4.10 | Iterated CKF estimation of accelerometer and gyroscope bias with dataset 2011_09_30_0034 | 89 |
| 4.11 | Experimental results presented in Google map for 2011_09_30_0033 | 90 |

| | | |
|------|---|-----|
| 4.12 | RMSE in the experiment with dataset 2011_09_30_0033 | 90 |
| 4.13 | Rotation errors in the experiment with dataset 2011_09_30_0033 | 90 |
| 4.14 | Illustration of the vehicle in wireless communication process for pseudo-range measurements | 92 |
| 4.15 | System architecture for integrating pseudo-ranging measurement | 94 |
| 4.16 | Position estimation of dataset 2011_09_30_0034 presented on Google map . | 96 |
| 4.17 | RMSE evaluation of position estimation between these solutions | 96 |
| 4.18 | Processing time evaluation of these solutions | 97 |
| 5.1 | Parallel processing for each feature | 101 |
| 5.2 | The architecture of a multi-core CPU | 113 |
| 5.3 | The proposed VIO system architecture where the gray blocks are executed in parallel on a multi-core CPU | 114 |
| 5.4 | The iteration process for feature f measurement update on one core | 115 |
| 5.5 | Speed-up and efficiency evaluation on laptop and mini PC kit without optimization process | 118 |
| 5.6 | Speed-up and efficiency evaluation on laptop and mini PC kit with optimization process | 119 |
| 5.7 | Speed-up and efficiency evaluation on desktop | 119 |
| 5.8 | Position estimates of the KITTI dataset 2011_09_30_0034 | 121 |
| 5.9 | RMSE evaluation for dataset 2011_09_30_0034 | 122 |
| 5.10 | Rotation error evaluation for dataset 2011_09_30_0034 | 122 |
| 5.11 | Iterated CIF estimation of velocity with dataset 2011_09_30_0034 | 122 |

| | |
|---|-----|
| 5.12 Iterated CIF estimation of the accelerometer bias and gyroscope bias with 2011_09_30_0034 | 123 |
| 5.13 Position estimates of the KITTI dataset 2011_09_30_0033 | 124 |
| 5.14 RMSE evaluation for dataset 2011_09_30_0033 | 124 |
| 5.15 Rotation error evaluation for dataset 2011_09_30_0033 | 124 |
| 5.16 RMSE evaluation for 11 sequences of EuRoC dataset [26] | 125 |
| 5.17 Experimental results of EuRoC dataset V1_02_medium with RMSE eval- uation of positional estimation | 126 |
| 5.18 Experimental results of EuRoC dataset MH_04_difficult with RMSE eval- uation of positional estimation | 127 |

Abbreviations

| | |
|--------|--|
| ASIC | Application-Specific Integrated Circuits |
| AV | Autonomous Vehicle |
| CIF | Cubature Information Filter |
| CKF | Cubature Kalman Filter |
| CPU | Central Processing Unit |
| EKF | Extended Kalman Filter |
| FPGA | Field Programmable Gate Arrays |
| GPS | Global Positioning System |
| GPU | Graphics Processing Unit |
| IEKF | Invariant Extended Kalman Filter |
| IMU | Inertial Measurement Unit |
| KF | Kalman Filter |
| MAV | Micro Aerial Vehicle |
| MSCKF | Multi-State Constraint Kalman Filter |
| MLE | Maximum Likelihood Estimate |
| RANSAC | RANdom SAmple Consensus |
| RMSE | Root Mean Square Error |

| | |
|------|---------------------------------------|
| SLAM | Simultaneous Localization and Mapping |
| TTG | Trifocal Tensor Geometry |
| UKF | Unscented Kalman Filter |
| VINS | Visual Inertial Navigation System |
| VIO | Visual Inertial Odometry |

Chapter 1

Introduction

This chapter begins by providing motivation for the research of autonomous robot navigation in GPS-denied environments. The chapter provides an overview of some available localization techniques. The chapter ends by providing the research statement, and the expected contributions of the research described in this thesis.

1.1 Motivation - Autonomous Vehicles

In recent years, Autonomous Vehicles (AV) have seen increasing demand in many industrial setting to replace traditional human-operated services. In particular Micro Aerial Vehicle (MAV) type drones have taken a special role in many industrial and service sectors. An excellent example is the employment of the MAV on an offshore oil-and-gas platform to perform structural visual inspection of the live flare, chimney and under-deck. There are many companies providing tele-operated drone based inspection, and such examples are Cyberhawk [2], Microdrones [7], and The Sky Guys [11]. MAV based inspection has improved the effectiveness and efficiency of inspections and allows performance of those

inspections without shutting down the system. The possibility of deploying MAVs with

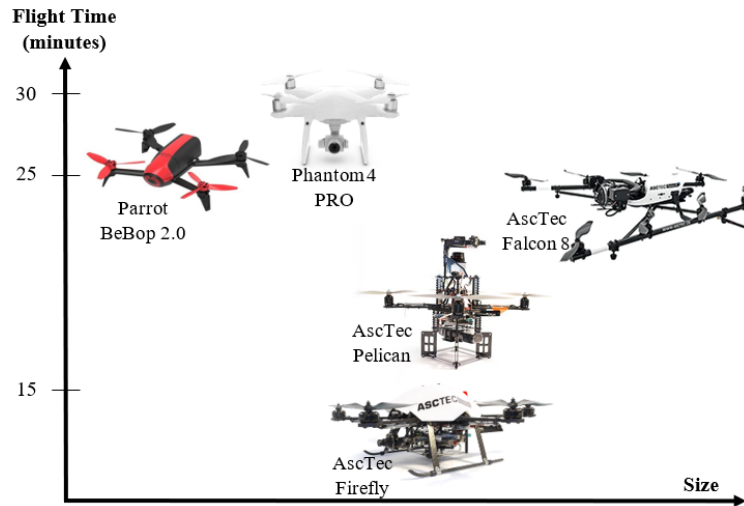


Figure 1.1: Some popular MAV's flight time for manual operation

full autonomous capability is still questionable due to many technological shortcomings. Inspection on the offshore platforms are enhanced in constrained areas and specific environments where Global Positioning System (**GPS**) signals are not available or reliable. Alternative navigation solution is required to help the MAV localize in the working environment. The employment of MAVs also gives rise to the issue of energy consumption, which limits the flight time. Figure 1.1 presents some popular commercial MAVs with their flight times. Recently, the maximum flight time is about 30 minutes for manual operation. On the other hand, fully autonomous MAVs are required to perform additional processing of data and computer algorithms such as localization, obstacle avoidance, automatic control, optimal trajectory generation for autonomous navigation. These algorithms require more computational resources, which in turn reduce the flight time significantly [125, 134]. With the limited time, this is a question as to whether MAV deployment is still beneficial and

efficient for the industry? It is also difficult for micro- and nano- robotic systems, having a limited payload and hardware computing capability, to operate autonomously. These shortcomings demand the development of a navigation technique in GPS-denied environments with affordable computational cost and processing time.

1.2 Autonomous Navigation in GPS-denied Environment

1.2.1 Visual Inertial Navigation System

During recent years, multiple navigation strategies (or technologies) have been investigated: wireless sensor network based localization [75], inter-robot relative measurement [42], a motion capture system [91], as well as a visual fiducial system [106]. These strategies utilize transmitter-receiver wireless communication or visual marker tracking to conduct relative measurements and track the vehicle trajectory. Hence, it limits the operating space of the vehicle and requires the installation of stationary sensor nodes in the working environment.

However, self-localization without GPS signals can be achieved using a camera image, which contains high-dimensional measurements and rich information about the environment. One popular strategy to utilize the camera image for navigation is Simultaneous Localization and Mapping (**SLAM**) [49, 77, 100]. Visual SLAM attempts to estimate both the vehicle trajectory as well as the location of environmental landmarks with respect to a navigation frame (Fig.1.2). SLAM utilizes optimization methods to improve the state estimation accuracy while generating the map and also re-localizing when the vehicle makes a loop closure. Multiple visual SLAM-based systems have been developed for a variety

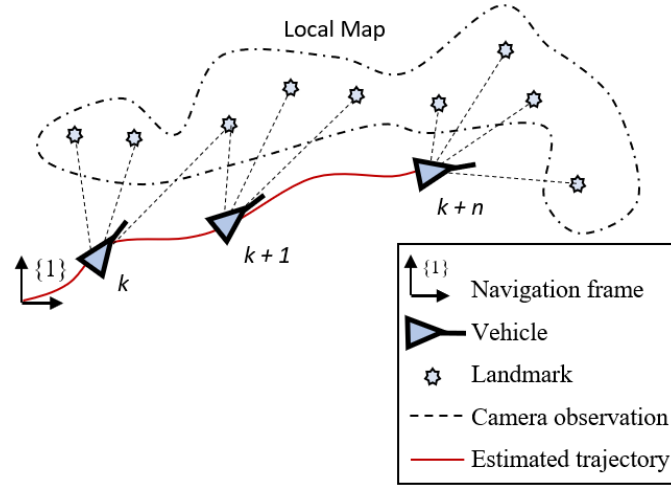


Figure 1.2: Visual SLAM application

of applications in GPS-denied environments. However, the major issue of SLAM is that it requires a considerable amount of computational resources and larger memory space. Running SLAM in hardware constrained systems is expensive and limits the applications for long-term operations. In particular, the application of SLAM for small-scale MAV is not an optimal solution. Additionally, the map building in SLAM is not always necessary if the objective is to find the state which including the position and orientation of the vehicle.

Alternatively, Visual Inertial Odometry (**VIO**) can track the vehicle trajectory without building the map using the camera images and Inertial Measurement Unit (**IMU**). This sensor suite is a common choice for autonomous vehicles and robots because of its light weight and reasonable cost. VIO navigation can be deployed as a stand-alone self-localization solution or as a part of larger localization and surveying application in GPS-denied environments. The technique is also useful for indoor navigation, virtual reality and augmented-reality applications. Some commercial VIO developer kits have been developed such as Optor Visual Inertial Camera [9], Parrot S.L.A.M. DUNK [10], Intel[®] Euclid[™] [4], and

Intel[®] RealSense[™] ZR300 [5]. Different camera models have been exploited in the system design such as RGB-D [72, 139], stereo [38, 138], monocular [81, 95] and fisheye [27, 120].

The literature of visual inertial sensor fusion is vast with an increasing number of open-source projects in recent years. Table 1.1 introduces some open-source VIO developments receiving great attention from the robotic research community. In general, these sensor fusion algorithms compromise two main functions: optimization and filtering. The optimization function attempts to construct the sensor fusion as a nonlinear optimization problem to extract the camera trajectories and 3D landmarks. The optimization process can achieve the highest accuracy but demands considerable computational resources. In contrast, the filtering function utilizes a nonlinear Kalman filter to restrict the inference process to the latest state of the system and represents the uncertainty by the covariance matrix [53]. This thesis describes the research which includes the filtering approach to implement a fast VIO system.

Table 1.1: Notable open-source VIO (regular type) and SLAM-based (*italic type*) developments in recent years

| Year | VIO Developments |
|-------------|---|
| 2007 | MSCKF1.0 [95] |
| 2012 | <i>PTAM</i> [49] |
| 2013 | MSCKF2.0 [79], <i>OKVIS</i> [78] |
| 2014 | SVO [54], <i>LSD-SLAM</i> [48] |
| 2015 | ROVIO [24], REBiVO [133], <i>ORB-SLAM</i> [100] |
| 2016 | Duo-VIO [41], Kalibr [122] |
| 2017 | VI-MEAN [150], VINS-Mono [119], <i>ORB-SLAM2</i> [101], <i>VIORB</i> [102], <i>S-PTAM</i> [116] |
| 2018 | Maplab [128], VI-DSO [141], Stereo MSCKF [132], <i>ICE-BA</i> [83] |

1.2.2 Issues Related to Visual Inertial Odometry Filtering

In general, the development of a VIO system should address some common technical issues. A brief insight into each issue is summarized below:

Feature-based image processing: The system operates on the observation of the landmark features in the working environment, which reveals the ego-motion of the vehicle [95, 124]. The observation of the same feature in multiple consecutive camera images results in geometric constraints between these camera poses. These constraints can be used to perform filter state correction and improve positioning accuracy. Algorithms for feature detection and matching are applied to track the feature through multiple camera image frames. Environmental noise and the presence of moving objects in the dynamic environment can also negatively affect the tracking process, which in turn decreases estimation accuracy. The number of tracked features is also important when considering the performance of the visual measurement update. The low density of landmark features can result in a small number of geometric constraints, which makes the system ineffective in eliminating the measurement noise.

Sensor fusion algorithm: VIO is developed to track the vehicle's ego-motion while capturing the benefits of both the visual and inertial measurements. Although a camera image contains high-dimensional measurements and important information about the environment, its update-rate is quite low ($\sim 20\text{Hz}$) which consequently slows the estimation process. On the other hand, IMU measurements can track the vehicle trajectory with a higher update-rate (100-200Hz), but with lower accuracy and a considerable amount of uncertainty. Sensor-fusion attempts to address robust and accurate localization while capturing the benefits of both the sensors. The hybrid solution can resolve the scale ambiguity

of monocular vision, observe gravity effects and produce reliable inter-frame motion estimates for more aggressive and agile trajectories. In general, VIO is a highly nonlinear problem to be resolved. The strategy to handle the system nonlinearity affects the estimation error and robustness of the performance. In the scheme of a Bayesian filter with Gaussian approximation, the Extended Kalman Filter (**EKF**) is a popular nonlinear filtering solution for VIO sensor fusion [79, 95]. However, it is well known that the first linearization in the EKF formulation causes several issues related to the system's accuracy and stability, particularly in dynamic environments. These limitations affect the quality of VIO estimation negatively.

Hardware implementation: VIO demands heavy computation associated with image processing and sensor-fusion at each time instant. This makes hardware implementation quite challenging and less trivial for micro- and nano- robotic systems. It is impractical to perform an entire-trajectory optimization due to the heavy computational load and processing time. The execution needs to be fast enough to keep track of the vehicle trajectory. To improve the computational efficiency, in many cases it is necessary to scale down the VIO system and select a small subset of tracked landmark features. Specialized hardware solutions with parallel processing capability such as Field Programmable Gate Arrays (**FPGA**), Graphics Processing Units (**GPU**) and Application-Specific Integrated Circuits (**ASIC**) can help to speed up the execution significantly [82, 126, 131, 157]. Additionally, the advent of non-von Neumann computing architectures and devices, such as the neuromorphic processing unit and quantum annealers [59, 117, 156], have shown the potential to perform complex computation faster and more power-efficiently than traditional processors. However, these specialized hardware computing platforms are some of less mature technologies and the scope of using them for real-time applications is less likely in the near future. The hard-

ware development for perception also requires a costly and time-consuming process. Even if such hardware is available, the system needs to execute identical computational steps at high frequency. This results in a decrease in processing time, but an increase in energy consumption. This is not beneficial for micro robotic systems with limitations on payload and energy storage. It may therefore be a better strategy to accelerate the computation using an available hardware configuration.

1.3 Thesis Problem Statement

The main objective of the research described in this thesis is the development of a fast and accurate visual inertial odometric computation methodology while maintaining comparative estimation accuracy over long-term operation. Many solutions are proposed to speed up the execution and address the following key problems:

1.3.1 Problem I: Computational Cost of Visual Measurement Model

Traditional VIO demands a considerable amount of computational resources to process a large amount of visual data. Such methods also require the reconstruction of 3D feature points using multiple camera frames. For example, traditional VIO [95] includes 30 camera poses in its filter state to improve the reconstruction accuracy. In general VIO systems nearly 50+ feature points are tracked to correct the predicted state optimally. Hence, the system needs to resolve 50+ optimization problems prior to completing each filter update step. The reconstruction process consumes a considerable amount of computational resources and processing time, which makes hardware implementation quite challenging. Also, this step limits the execution of other advanced autonomous functions such as map-

ping, object detection and fuel-optimal path planning. Especially for hardware limited robotic systems, such as micro- and nano- robot applications, it is necessary to develop a computationally-efficient visual measurement model for fast implementation [43, 71, 80]. Trifocal Tensor Geometry (**TTG**) [60] has been introduced as an efficient approach to perform point transfer across three consecutive camera frames. This thesis describes the research which applies the TTG-based approach to predict the feature point measurements and replace the expensive conventional approach.

1.3.2 Problem II: Nonlinear Filtering for Visual Inertial Odometry

Nonlinear systems require nonlinear sensor fusion algorithms within the new visual measurement model. This sensor fusion algorithm should not increase the computational cost significantly, and should achieve a similar or better estimation accuracy. As the research described in this thesis focuses on the inference of the vehicle's latest state rather than the entire trajectory or the construction of a map, the sensor-fusion algorithm follows nonlinear Kalman filtering. Geometric constraints from multiple camera views are exploited to design the visual measurement update step.

Cubature Kalman Filter (**CKF**) [18] is initially applied for VIO applications. In the literature of Kalman filtering state estimation, CKF has shown great potential to replace EKF to handle system nonlinearity. Instead of linearizing the nonlinear models, these filters generate a minimal set of sample points to propagate through the nonlinear functions. This strategy produces better estimation of the mean and covariance [18].

The research described in this thesis also addresses the VIO computational efficiency associated with Kalman filtering structure using a Cubature Information Filter (**CIF**) [109].

CIF execution avoids the inverse computation of the high-dimensional innovation covariance matrix, which in turn further improves computational efficiency of the VIO system.

1.3.3 Problem III: Error Accumulation over Long-term Operation

The VIO system suffers from error accumulation; one correction from a single measurement cannot effectively eliminate the accumulative error existing in the camera observations. This limitation makes the filter gradually diverge and even fails to track the vehicle trajectory over long-term operation. The VIO only produces a reliable estimation of the vehicle trajectory in short-term operation and short-distance travel. This issue demands the development of a VIO technique to achieve a bounded estimation error over long-term operation. Some advanced techniques, such as global pose graph optimization [82] and loop-closure [78], require relatively high computation and processing times to execute the optimization procedure for the entire trajectory. They also may not be feasible to be implemented in a low-cost flying platform.

In an attempt to allow the VIO to operate for a longer duration in the absence of any map, the thesis describes research which examining two filtering solutions. The first solution implements iterated CKF, which performs multiple corrections on a single measurement to optimize the latest filter state and covariance during the measurement update. The optimization process is terminated using the Maximum Likelihood Estimate (**MLE**) based criteria. The second solution integrates CKF with pseudo-ranging measurement between the vehicle and multiple beacons. The integration follows the sequential-sensor-update approach, which in turn enables independent operation between ranging sensors and VIO system.

1.3.4 Problem IV: Parallel Processing to Accelerate the Computation

Conventional filtering systems are inefficient when processing massive amounts of visual data in a serial manner during the filter measurement update. It is impractical to deploy an additional optimization execution, which increases the computational load and processing time significantly. Parallel processing with specialized hardware, such as FPGA [157], GPU [82, 126, 151] and ASICs [131], can help to accelerate the computation effectively. However, the development of these specialized hardware computing platforms for perception is an expensive and time-consuming process. Also, the resulting hardware is difficult to upgrade [28]. Alternatively, the multi-core CPU has become more popular and also generally available in all regular computers. This solution was selected to enhance parallel processing and accelerate execution.

This thesis initially implements parallel processing for VIO applications using a multi-core CPU. We distribute the computation of the visual measurement update by strictly assigning the computational task to each core. The distribution exploits the cubature information filter structure and assumes that each visual feature has a different contribution to the motion estimation. Furthermore, the system architecture is able to execute maximum likelihood estimation based optimization for individual feature measurement, which helps to minimize error accumulation over long-term operations.

1.4 Objectives and Expected Contributions

Objective 1: Design a nonlinear filtering algorithm using trifocal tensor geometry and cubature Kalman filter.

- Contribution 1: This thesis presents the development of a feature point transfer approach using TTG to replace the 3D feature point reconstruction step in the traditional VIO. The replacement helps simplify the system architecture and speed up the estimation.
- Contribution 2: Experimental comparisons between two measurement models help to determine the benefit of using TTG in VIO application.
- Contribution 3: A cubature Kalman filter with a spherical deterministic sampling approach is developed for VIO application to handle the nonlinearity effectively.
- Contribution 4: A cubature information filter is developed for VIO applications. Scheduling the computation on the information domain helps to address the system efficiency issue associated with Kalman structure.

Objective 2: Address the VIO estimation's error accumulation over long-term operation.

- Contribution 5: Utilize the benefit of TTG based measurement model as a non-recursive function to enhance the optimization during the filter update step. This strategy does not increase the system complexity significantly or require the installation of any advanced optimization library, which may not be applicable for some particular robotic systems. This solution is suitable for self-localization projects.
- Contribution 6: Combine VIO and pseudo-ranging measurements to bound the estimation error over long-term operation. The integration is enhanced with a sequential-update approach following the standard EKF structure. This solution can be applied for larger navigation projects.

Objective 3: Enhance parallel processing for VIO system with multi-core CPUs.

- Contribution 7: The proposed system architecture utilizes multiple cores of a generic computer for parallel processing. It speeds up the estimation significantly without the installation of specialized hardware computing platforms.
- Contribution 8: A cubature information filter is extended to the field of invariant Kalman filtering, and then applied to a VIO system. This combination helps to improve the estimation accuracy and consistency. The information structure also helps to distribute the computation effectively to multiple cores.
- Contribution 9: Optimization is also employed to minimize the error accumulation over long-term operation. This procedure is applied independently for individual features during parallel processing.
- Contribution 10: The experiments use various multiple-core hardware platforms with a different number of core and computing capabilities for validation. This helps to evaluate the efficiency when deploying more cores to accelerate the computation.

1.5 Organization of the Thesis

This thesis presents research across three different problem areas related to visual inertial odometry. A brief outline of each chapter is described as follows:

Chapter 1 - Introduction: This chapter introduces the research motivation, the considered systems, their associated problems and the main objectives of this thesis, as well as the proposed solutions.

Chapter 2 - Background and Literature Review: This chapter provides a brief background of visual inertial odometry. The main drawbacks of some available solutions are

discussed, which helps to determine the main challenges for the algorithm development.

Chapter 3 - Developing Computationally-Efficient Nonlinear Cubature Kalman Filtering for Visual Inertial Odometry: This chapter addresses objective 1 of the thesis. A trifocal tensor geometry based measurement model is developed and compared with the conventional model. This chapter describes in detail the development of a cubature Kalman filter for VIO sensor fusion.

The content of this chapter is based on the following papers of the author:

- T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "Developing a Cubature Multi-State Constraint Kalman Filter for Visual-Inertial Navigation System", Canadian Conference on Computer and Robot Vision, May 2017
- T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "Developing Computationally-Efficient Nonlinear Cubature Kalman Filtering for Visual Inertial Odometry", ASME Journal of Dynamic Systems, Measurement and Control, February 2019

Chapter 4 - Accurate Visual Inertial Odometry for Long-Term Trajectory Operations without Using a Map This chapter relates to objective 2 of the thesis by proposing two solutions. Solution 1 employs iterated CKF, which performs optimization during the visual measurement update step. Solution 2 integrates VIO estimation with pseudo ranging measurements.

The content of this chapter is based on the following papers of the author:

- T. Nguyen, G. K. I. Mann, A. Vardy, and R. G. Gosine, "Likelihood-based Iterated Cubature Multi-State-Constraint Kalman Filter for Visual Inertial Navigation System", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), October 2017

- T. Nguyen, G. K. I. Mann, A. Vardy, and R. G. Gosine, "CKF-Based Visual Inertial Odometry for Long-Term Trajectory Operations", ASME Journal of Dynamic Systems, Measurement and Control (Under Review), 2019

Chapter 5 - Efficient Parallel Processing Solution to Accelerate Visual Inertial Odometry Execution on a Generic Computer: This chapter relates to objective 3 of the thesis. It details the deployment of parallel processing on a multi-core CPU and experiments to evaluate the efficiency of the deployment.

The content of this chapter is based on the following paper of the author:

- T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "Accelerating Visual Inertial Odometry Using Parallel Processing and Cubature Information Filter", IEEE/ASME Transactions on Mechatronics, (Under Review), 2019

Chapter 6 - Conclusion: This chapter concludes the thesis by discussing the application scenarios of the proposed VIO algorithm, its drawbacks, and potential topics that require further investigation.

Chapter 2

Background and Literature Review

This chapter presents background literature related to visual inertial odometry that is relevant to the research. The main issues associated with the sensor fusion problem are presented along with how they are addressed by research described in the literature.

2.1 Overview of Visual Inertial Odometry

Initial approaches to fusing IMU measurements and camera images are reported in many projects [22,55,118,123]. Motion estimation is often developed using only two consecutive camera frames. For example, the researches [118,123] estimated the visual displacement between the current and previous image before combining with inertial measurements using an EKF. Similarly, Diel et al. [45] built the visual measurement update on the epipolar geometry constraint between two images. Initial VIO developments also include multiple SLAM-based systems as reported in [74,107,130]. They exploit the correlations between multiple camera poses and 3D positions of the observed visual landmarks to produce accurate localization results. The use of multiple camera frames helps to achieve higher esti-

mation accuracy. However, performing visual SLAM in the environment with thousands of landmark features is challenging for real-time implementation [95]. This limitation gives rise to the need for an alternative sensor-fusion algorithm to accommodate the high volume of data processing for real-time operation while satisfying the requirement of estimation accuracy.

2.1.1 General System Architecture Design

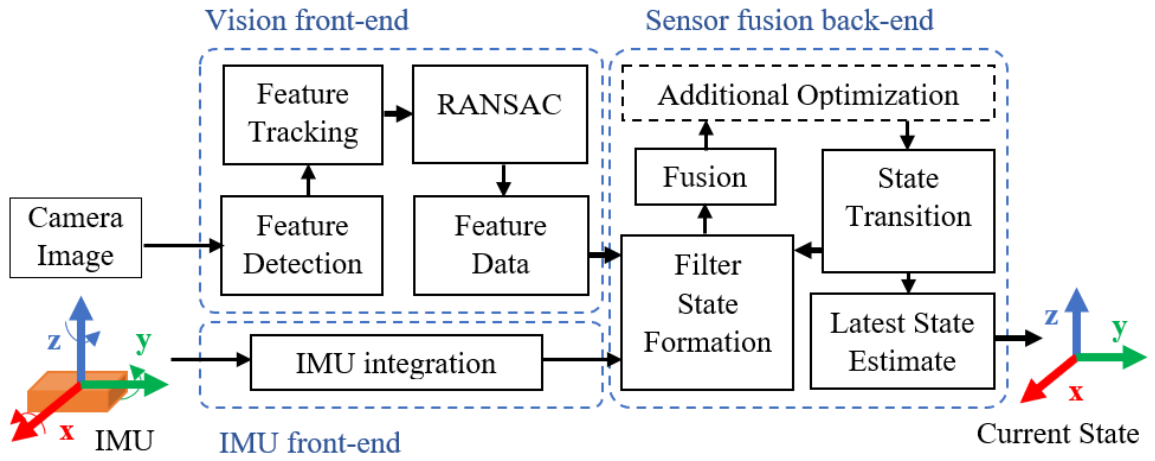


Figure 2.1: General tightly-coupled VIO system architecture

Generally, VIO sensor fusion utilizes either loosely-coupled [127] or tightly-coupled integration [95]. Loosely-coupled systems produce the odometry estimation after processing IMU data and camera observations individually. This decoupling decreases the system complexity but produces poor estimates due to the inability to estimate IMU biases [39]. Consequently, loosely-coupled systems cannot achieve the estimation accuracy and consistency of the tightly-coupled system, which processes all information together. Figure 2.1 demonstrates an overview of the general architecture of tightly-coupled systems with mul-

multiple critical parts. The front-end and back-end parts are designed to exploit the benefits of both sensors, capture the system uncertainty, produce reliable ego-motion estimation and achieve robust performance.

Usually, the VIO system is designed to track IMU frames. The IMU front-end utilizes the high-frequency outputs of the gyroscope and accelerometer to predict the relative pose of the current frame with respect to the previous frame. This IMU prediction, which has low accuracy and a considerable amount of uncertainty, is corrected by the camera observations of 3D feature points in the environment. The vision's front-end focuses on tracking these features. Multiple approaches have been applied for feature detection such as Shi-Tomasi [71, 82, 129], Scale-Invariant Feature Transform (**SIFT**) [64, 88, 95], Speeded Up Robust Features (**SURF**) [21, 39, 110] and Feature from Accelerated Segment Test (**FAST**) [132, 136]. These approaches aim to detect some specific features, which are distinctive in the environment and robust in the cases of different lighting and presence of visual noise. These features are tracked temporally using the KLT optical flow algorithm [89, 132] or feature descriptor matching [64, 88, 110]. RANdom SAMple Consensus (**RANSAC**) is also applied to reject outliers in temporal tracking. Inliers are reliable feature points, which will be utilized in the sensor-fusion back-end. Better selection of the feature detection and matching can help to reduce the computational requirements.

Not all features have the same contribution to motion estimation [28, 159]. Motion blur and various texture gradients can cause the feature observations to be less informative to the filter update [114]. In reality, most VIO algorithms make a strong assumption about all visual features having equal priority for the filter operation. The system constructs all of them in a single composite group measurement model. However, this simplification results in a higher-dimensional matrix of the measurement model and innovation covari-

ance for computation [31, 33, 103, 109]. Peretroukhin et al. [112, 114] have eliminated the static assumptions of observation uncertainty parameters. An observer is developed to evaluate the qualitative contributions of each feature. The training phase is quite challenging to produce the best model which effectively captures the influence of individual features on the estimation. Beside tracking visual landmark features, some projects focus on image-intensity [160] and the straight-line feature [152, 153]. These strategies are useful for some particular environments such as corridor operation, fast-motion and low-light situation where the feature points are limited and unreliable to track. This thesis only focuses on visual features.

The sensor fusion back-end is the main focus of this thesis. Recent literature has introduced multiple publicly available datasets supporting the development of a VIO sensor fusion algorithm. These datasets help to validate the VIO algorithm in practical conditions with noises and moving objects. Also, using the same dataset encourages a rigorous comparison between VIO algorithms, which helps to determine the benefits and limitations of each algorithm. For this purpose, recent literature has introduced multiple datasets (Table 2.1). Notably, KITTI [56] and EuRoC [26] have received special attention due to their good documentation and ease of use. The EuRoC dataset [26] was collected by MAV in robotic laboratory and factory, where the VIO can be verified with visual inspection activities. The KITTI dataset [56] was collected while driving a car in city and residential areas, which provides a basic test to the algorithm for long-term operation and long-distance travel. The research described in this thesis utilizes these two datasets for experimental validation.

Table 2.1: Overview of publicly-available datasets

| | KITTI [56] | EuRoC [26] | PennCOSYVIO [115] | Canoe [92] |
|------------------|-------------------------------|-------------------|--|-------------------|
| Year | 2012 | 2016 | 2016 | 2017 |
| Environment | Outdoor | Indoor | In/outdoor | Outdoor |
| Carrier | Car | MAV | Hand-held device | Canoe |
| Scene | City-scale | Factory/Lab | Campus-scale | River |
| Distance (total) | ~39km | ~800m | ~600m | ~2.7km |
| Camera | 1 stereo RGB 1 stereo gray | 1 stereo gray | 4 RGB 1 stereo gray 1 fisheye gray | 1 stereo RGB |
| IMU | 1 accel/gyro | 1 accel/gyro | 2 accel, 2 gyro 1 acc/gyro | 1 accel/gyro |
| GPS | 1 | none | none | 1 |
| Ground-truth | GPS/INS | MoCap/Laser | Visual tags | GPS/INS |
| Accuracy | ~dm | ~mm | ~dm | ~dm |

2.1.2 Classification of Tightly-coupled VIO Algorithms

Although the VIO literature is vast, for the purpose of this research the most relevant papers can be considered along two directions as shown in Table 2.2. The first direction corresponds to the theoretical developments and practical implementation. It results in three groups of relevant literature. Group A completely focuses on developing novel VIO algorithms from a theoretical viewpoint, which are implemented on a powerful computer. These findings construct better strategies for approximating the nonlinear model, sensor-fusing and capturing the uncertainty, which significantly improves the positioning accuracy. Groups B and C consider the limitations of the hardware computing platform in their developments. Group B addresses the issue through a computationally-efficient algorithm, which uses fewer resources but still satisfies the estimation accuracy. In group C, the contributions focus entirely on the specialized hardware development to accelerate the computation. More discussions of group B and C will be presented in section 2.3.

The second direction considers the number of camera poses contributing to the estimation:

- **Full-smoothing:** This approach attempts to estimate the entire trajectory, all camera poses and 3D landmarks, as a nonlinear optimization problem. This process can guarantee the highest accuracy but requires heavy computational efforts [53, 65, 67]. In real-time operation, all the information such as the map and trajectory will grow quickly, which makes the implementation infeasible. Selecting useful keyframes [82, 104] or parallel execution [69, 96] for the optimization can help to accommodate the heavy computational load [53].
- **Fixed-lag smoothing:** The fixed-lag smoothing approach is a compromise between

accuracy and computing efficiency, limiting the optimization within a sliding window and marginalizing older camera poses [46, 63, 71, 78]. When setting up the system using maximum likelihood estimation, the optimization problem is applied over a set of recent states [23, 71]. However, determining an appropriate number for the window length makes it challenging to meet an expected level of performance.

- **Filtering:** In contrast to these optimization based approaches, this subgroup restricts the inference process to the latest state of the system and represents the uncertainty by the covariance matrix. In practice, this approach can be upgraded to the fixed-lag smoothing approach to achieve better estimation accuracy [53, 71] because it re-linearizes past measurements during the smoothing process. The research described in this thesis develops the sensor fusion algorithm following the filtering subgroup to produce a fast VIO estimation. Some related researches will be discussed in section 2.2.

Table 2.2: The most relevant VIO papers considered in this thesis

| Group | A | B | C |
|---------------------|-----------------------------------|--------------|-----------|
| Filtering | [25, 28, 39, 64, 79, 87, 95, 132] | [50, 58, 80] | [44, 157] |
| Fixed-lag smoothing | [46, 78] | [63, 71] | [41, 124] |
| Full smoothing | [52, 102] | [82] | [28, 131] |
| VIO augmentation | [73, 85, 86, 112–114] | | |

2.2 Filtering Methods for VIO Algorithm

A typical example of the filtering subgroup is the multi-state constraint Kalman filter [79, 95]. In general, MSCKF operates the standard extended Kalman filter over a sliding window of multiple subsequent camera states [125]. Figure 2.2 illustrates the data updating in this VIO system with a sliding window. The camera state has the uncertainty as prior information. The filter estimation exploits the geometric constraints through the observation of an identical feature in different images. The relative motion constraints between camera states are also utilized using the IMU measurements in the filter prediction. In the visual measurement update, an inverse-depth least-squares optimization technique is applied to compute the 3D landmark feature points using multiple camera states and camera measurements over a defined sliding window [39, 95]. Then, the re-projection errors are calculated to correct the predicted filter state [95]. This process does not account for the prior information and relative motion constraints. Consequently, the computational cost and system complexity are maintained at a lower level relative to the visual SLAM-based technique, which entirely exploits all the available information [124].

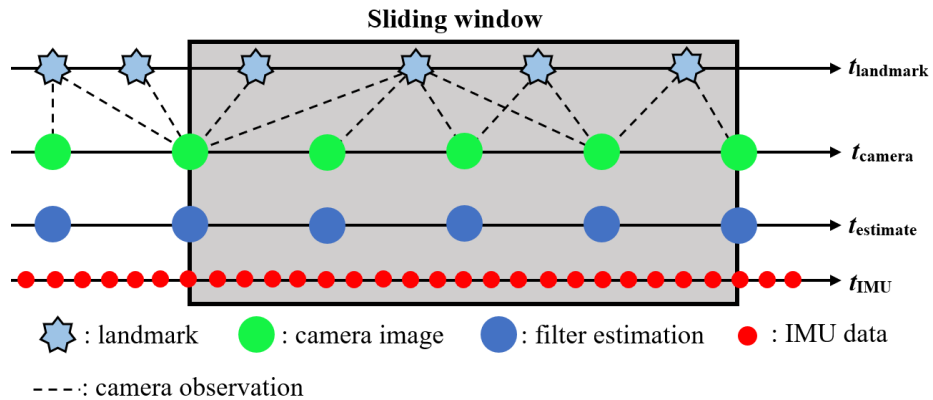


Figure 2.2: Illustration of the data updating in VIO system

MSCKF shows poor estimation accuracy and consistency, inherited from EKF implementation. Also, EKF-based VIO systems have a limited capability to handle unobservable transformations (i.e., three Degrees of Freedom (**DoF**) global translation and one-DoF rotation about the gravitational vector) [61]. Hesch et al. [61,62] have addressed the issue using observability-constrained EKF, for which the Jacobian matrices were modified to explicitly enforce the un-observability along specific directions. Bonnabel et al. [19,20] proposed the EKF modification for nonlinear systems possessing symmetries (or invariances) to address the inconsistent issue of the traditional EKF. Their invariant EKF is a combination of the symmetry-preserving observer and EKF. Also, observability analyses [20, 148] have indicated the absence of the invariance affects negatively to the consistency of the MSCKF estimates. Wu et al. [148] have initially applied invariant EKF for VIO application to improve the estimation consistency. However, these invariant filters still inherit the EKF's limitation in handling the system nonlinearity.

Alternatively, multiple solutions have been proposed to replace EKF by other advanced nonlinear Kalman filters. The Unscented Kalman Filter (**UKF**) [142] was introduced as a Jacobian-free filter, using a deterministic sampling technique to handle the nonlinearity. Unlike a stochastic sampling technique employed in particle filter [40], the technique carefully selects a minimal set of sigma points with a defined pattern to completely capture the true mean and covariance of the Gaussian random variables. These sigma points are propagated through nonlinear functions to capture the posterior mean and covariance accurately. This strategy can handle the nonlinearity with better estimate of mean and covariance. Some researches [64, 70, 87] have examined UKF with VIO application and received positive results. However, the drawbacks of UKF are in the complicated implementation. For example, the computed covariance matrix is not always guaranteed to be

positive definite for square-root operation because of the presence of negative sigma points. This issue may cause the filter to halt the VIO system functioning. The Cholesky factorization approach could be applied to satisfy that condition, but it is shown to be inefficient and not always applicable for all the cases [18]. Arasaratnam et al. [18] overcame this issue using a spherical-radial cubature rule. Their cubature Kalman filter execution better handles the negative sigma points while improving the system accuracy and stability. Literature also introduces the mathematical transformation of the Kalman filter to the information domain, which improves the computational efficiency [103]. The transformation is suitable for some complex systems with high-dimensional measurement models or multiple measurement updates [103, 109, 145].

Meanwhile, some researches [23, 24, 149, 155] attempt to reformulate VIO with respect to a local moving frame, instead of using the standard world-centric formulation. The modification has relaxed the requirement to align the initial orientation with the global gravitational direction. The filter estimates the landmark 3D position in the local frame, which in turn makes the system fully observable and avoids the inconsistency inherently. The disadvantages of this strategy are the penalty of larger uncertainty, a more complicated system to manage these tracked features and extra computations in the filter propagation, as discussed in [155]. In general, this robocentric strategy is still in the development process and requires more extensive verifications with different hardware computing platforms to determine its effectiveness.

VIO can exploit the geometric constraints between multiple camera views for the visual measurement update. The strategy to deploy these constraints depends on the application, which will decide the system complexity. By only using three camera views, Hartley et al. [60] introduced trifocal tensor geometry, which has been applied popularly in some

computer vision projects. The observation of the same feature in three consecutive images results in geometric constraints between three camera poses. These constraints can be used to produce control commands for visual-servoing [17]. The control law was simplified to enhance a fast trifocal control system while attaining global exponential stability and robust performance. In the application to assist a driver to see through the vehicle ahead for overtaking maneuvers, Rameau et al. [121] utilized TTG to filter out incorrect feature matching efficiently. This procedure also extracted the fundamental matrices and the camera trajectory to render the virtual objects. This process was similar to some visual odometry applications [34,93]. Moreover, their method [121] employed trifocal tensor image synthesis and marker-based pose estimation to generate a seamless transparency effect from the rear car's viewpoint. Their implementation reduced the quantity of information communicated between vehicles and achieved good real-time performances. Overall, the TTG employment is straightforward computation without any recursive execution. This benefit helps to reduce the computational cost of hardware implementation.

At the same time of this thesis, other projects are also attempting to use TTG in their VIO application, such as [64, 70]. Their initial results have confirmed the possibility of using TTG for developing a visual measurement model. The research described in this thesis conducts further analysis to determine the benefits of TTG employment relative to the traditional approach. Moreover, the research also proves that the simple design of a TTG based measurement model allows the implementation of additional optimization within the Kalman structure. Different Kalman filtering techniques are applied to handle the nonlinearity of measurement models such as the cubature Kalman filter, unscented Kalman filter, cubature information filter and unscented information filter. The use of each filter helps to improve the VIO performance in different aspects of implementation.

2.3 VIO Hardware Implementation

This section is an overview of the related researches in group B and group C, which are related to the VIO implementation on different hardware computing platforms.

2.3.1 Group B: Developing a Computationally-efficient VIO Algorithm

VIO demands heavy computation on image processing at each time instant, and this makes hardware implementation quite challenging and non-trivial. Any computing platform has limited processing and sensing capabilities, particularly those used for the embedded computing hardware of wearable devices [71], cellphones [58] and micro-aerial vehicles [87]. Often, proposed algorithms are modified so that on-line computation is feasible with minor losses in estimation accuracy. For example, Guo et al. [58] carefully scaled down the system dimension to almost half through the interpolation model in order to speed up the processing time significantly. Additionally, the QR decomposition implementation using the C++ Eigen library was able to reduce the computational complexity of calculating the Jacobian matrix.

As mentioned, the full-smoothing approach applies the optimization process for the entire trajectory, which quickly becomes infeasible as the trajectory and the map grow over time. A down-sample process was useful in this case to include only all primary keyframes within loop closure constraints. This process can maintain the pose graph database at a limited size but negatively affects the quality of the estimation. For example, in the research [44], the implementation of VINS-Mono without loop closure [119] on ODROID¹

¹ODROID is an embedded PC containing a hybrid processing unit. It is produced by Hardkernel Co., Ltd. [8] from South Korea

requires the reduction of the number of tracked features from 150 to 100. Delmerico et al. [44] also reported that VINS-Mono with loop closure cannot be achieved on ODROID [82]. Similarly, the implementation of OKVIS [78] on ODROID required the reduction of the maximum number of features from 400 to 200, the key-frame of the sliding window from 5 to 3 and the number of IMU linked frames from 3 to 2, reported in [44].

To improve the computational efficiency in many cases, the VIO process selects a small subset of tracked features. Carlone et al. [28] have developed a visual attention mechanism that helps to determine which feature is the most relevant visual cue to maximize the performance of the VIO system usefully. This strategy results in a smaller measurement model and innovation covariance matrix, which in turn reduces the computational efforts significantly. They also utilize an information filter [103] to process and evaluate the contribution of each feature to the motion estimate independently. The information filter is the mathematical transformation of the Kalman filter by taking the inverse of the covariance. Using the information filter is preferable for some systems having a large number of measurements, features or demanding a decentralized filter form [31, 103, 109, 145]. In the research described in this thesis, the information filter structure is utilized to distribute the computation to a multi-core CPU.

2.3.2 Group C: Accelerating Execution Using Specialized Hardware

Group C focuses entirely on specialized hardware development and optimal system architecture design. Nikolic et al. [105] implemented the execution of feature detection to FPGA, and was able to reduce the computational complexity significantly. The FPGA implementation also frees up computational resources for the optimization and other advanced

tasks of the SLAM system. Instead of partial deployment, Zhang et al. [157] have fully implemented the system in FPGA for execution. They have introduced an algorithm-and-hardware co-design approach to significantly speed up the estimation as well as optimize the energy consumption. Group C also includes the development of a perception module, which houses all computing and sensing modalities. This detachable module allows testing the VIO module independently or equipping for different robotic platforms. For example, some researches [82, 126, 151] developed a perception module for aerial robots' navigation. The computation was conducted on mini i7 computer and NVIDIA TX1. Notably, they utilized 256 NVIDIA CUDA GPU cores of NVIDIA TX1 to accelerate the computation of 3D feature points, depth images and global map generation. Furthermore, Suleiman et al. [131] fully integrated a VIO system into an ASIC based chip to eliminate costly off-chip processing and storage. This is strong evidence that parallel computing can speed up the computation efficiently with an affordable power budget. However, Carlone et al. [28] has determined some drawbacks associated with these specialized hardware platforms. First, the development of these specialized hardware consumes considerable investments and time. Second, the resulting hardware is challenging to upgrade in the future. Third, it may be desirable to develop a VIO framework which can systematically trade off performance for computation. The system performance is adjusted flexibly, depending on available computing resources and performance requirements in the working environment. These are the reasons why the research described in this thesis has decided to use a generic computer for VIO development. The availability and ease-of-use of multi-core CPU solution will help designers develop parallel processing capability for their own VIO system.

2.4 VIO Augmentation

The combination of VIO and other advanced sensors not only improves the long-term estimation accuracy but also the system performance in dynamic environments. A Light Detection and Ranging (**LIDAR**) sensor can help to reduce visual drift in undesired lighting conditions [66,154]. This solution can improve the positioning accuracy over long-term operation satisfactorily. However, the deployment of LIDAR can raise the issues of power consumption and payload for micro robotic systems.

Some research projects [73, 113] have measured the global orientation from the sun-direction estimation. Lambert et al. [73] utilized a sun sensor while Peretroukhin extracted directly from the existing image stream. The extraction was conducted by a sun detection model with a Bayesian convolution neural network. Although the estimation of a sun direction vector has improved vehicle trajectory tracking, that solution is not always available, particularly during cloudy weather and nighttime. It only affects the orientation and also requires considerable resources to train the sun detection model.

The wireless communication between the vehicle (tag) and known-location beacon (anchor) have been popularly applied to supplement the primary navigation system [37,66,98, 144, 146]. Such systems commonly detect the Time-Of-Arrival (**TOA**) of signals encoded in the radio or acoustic waves to conduct a ranging measurement. For example, some researches [99, 144] have deployed Ultra-WideBand (**UWB**) radio modules for ranging measurements. This technique suffers from systematic errors such as uncertain wave speed or clock synchronization errors. Consequently, it cannot directly measure the true geometric range, which is why it is called pseudo-ranging measurement [66]. By placing multiple anchors along the vehicle trajectory, the system has an additional source of measurement

to eliminate the drift of the VIO estimation. The effectiveness depends on the integration technique to align many estimates [29, 144].

2.5 Summary

This chapter presented many issues associated with the development of a VIO system. As shown in the literature, the VIO implementation requires a considerable amount of computational resources to process a large amount of visual data. The use of more powerful hardware computing platforms can accelerate the execution, but the process of hardware development is costly and time-consuming. The execution of the identical algorithm at a much higher rate also consumes more energy, which is not beneficial for micro-robotic systems. These limitations lead to the requirement of a computationally-efficient sensor-fusion algorithm.

The VIO literature also gives rise to the issue of error accumulation over long-term operation. Additional computations for minimizing the accumulative errors increase the computational cost significantly. The implementation also requires the installation of advanced computing libraries or significant modifications to the hardware system. This limitation motivates a better strategy to deliver that heavy computations such that the algorithm is still computationally-efficient and feasible for current hardware configurations.

Chapter 3

Developing Computationally-Efficient Nonlinear Cubature Kalman Filtering for Visual Inertial Odometry

This chapter¹ presents a computationally efficient sensor-fusion algorithm for visual inertial odometry. Trifocal tensor geometry is utilized for a visual measurement model and a nonlinear deterministic-sampling based filter, known as the cubature Kalman filter, to handle the system nonlinearity. The TTG-based approach is developed to replace the computationally-expensive 3D-feature-point reconstruction in the conventional VIO sys-

¹This chapter is based on the following publications of the author:

- T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "Developing a Cubature Multi-State Constraint Kalman Filter for Visual-Inertial Navigation System", Canadian Conference on Computer and Robot Vision, 2017
- T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "Developing Computationally-Efficient Nonlinear Cubature Kalman Filtering for Visual Inertial Odometry", ASME Journal of Dynamic Systems, Measurement and Control, February 2019

tem. This replacement has simplified the system architecture and reduced the processing time significantly. The CKF is formulated for the VIO problem, which helps to achieve a better estimation accuracy and a more robust performance than the conventional extended Kalman filter. Several experiments use the publicly-available datasets for validation and comparison with many other VIO algorithms available in the recent literature. Overall, this proposed algorithm can be implemented as a fast VIO solution for high-speed autonomous robotic systems.

3.1 Introduction

VIO application can range from autonomous ground robotic systems to unmanned aerial vehicles [43, 125], and with the increasing applications, the method is required to be computationally efficient for fast implementations. The traditional VIO demands a considerable amount of computational resources to process a large amount of visual data. This method also requires the reconstruction of 3D feature points using multiple camera frames. The reconstruction process consumes a considerable amount of processing time, which makes hardware implementation quite challenging. Especially for hardware limited robotic systems such as micro- and nano- robot applications, it is necessary to have an efficient VIO algorithm for fast implementation [43, 71, 80]. Trifocal tensor geometry [60] has been introduced as an efficient method for prediction of feature points in an image using three consecutive camera frames. This chapter will address this issue while implementing the TTG. The point transfer approach using TTG will replace the expensive conventional approach. A rigorous comparison between the two approaches helps determine the benefit of using TTG in VIO application.

The second contribution of this chapter is the development of the nonlinear filter for handling the TTG-based measurement model. Recently, multiple sensor-fusing algorithms have been introduced to improve estimation accuracy and robust performance in dynamic environments. A benchmark comparison between these publicly-available algorithms can be found in [44]. More recent developments of VIO algorithms, such as VINS-Mono [119] and OKVIS [78], have better estimation accuracy compared to the traditional MSCKF [79, 95]. However, these approaches require more computational resources to solve the optimization problem. MSCKF still plays an important role, since it has the ability to produce better consistency and update-rate among the variety of hardware computation platforms [44]. This chapter exploits this property to develop the TTG-based measurement model. MSCKF is a tightly-coupled extended Kalman filter based system operating over a sliding window of multiple subsequent filter states. Its poor estimation accuracy is inherited from the standard EKF implementation. Alternatively, this chapter will employ the cubature Kalman filter as a better solution to handle the system nonlinearity. CKF utilizes a deterministic sampling technique and cubature rule to generate a minimal set of sample points. Propagating these sampling points through the nonlinear functions results in a better approximation of the mean and covariance. This chapter also employs another version of CKF, known as the cubature information filter [109], to address the computationally-efficient issue of the Kalman filtering structure. When mathematically transforming CKF to the information domain, the execution avoids the computation of filter gain and the inverse of the high-dimensional innovation covariance matrix. Consequently, it improves the system architecture, which consumes less computational cost and evaluates the contribution of each feature measurement independently. To summarize, this chapter makes the following contributions.

- This chapter presents the development of a feature point transfer approach using TTG to replace the 3D feature point reconstruction step in the traditional VIO. The replacement helps to simplify the system architecture and speed up the estimation.
- CKF is applied to handle the system nonlinearity. To the best of our knowledge, this is the first employment of CKF for VIO application.
- CIF is developed to improve the overall computational cost when processing considerable amounts of visual data. Experiments use publicly-available datasets to validate the proposed algorithm as well as compare it to other advanced VIO algorithms in the literature.

The remainder of this chapter is organized as follows. The next section presents the VIO literature associated with the filtering approaches. Section 3.2 introduces the coordinate system, notation, and the IMU model. Section 3.3 describes the filter state propagation. Section 3.4 focuses on the TTG-based measurement model and compares it to the conventional model. Then, section 3.5 presents experimental validation for the proposed algorithm. Section 3.6 describes the formulation of CIF and its experimental validation. Finally, some conclusions are presented in section 3.7.

3.2 System Description

3.2.1 System Coordinates and Essential Notation

The coordinate frames of the system (Fig.3.1) are defined: $\{G\}$ is a global frame; $\{I\}$ is an IMU frame; $\{C\}$ is a camera frame. The navigation system consists of an IMU sensor

and a monocular front-looking camera. The transformation of $\{I\}$ with respect to $\{G\}$ is expressed by the translation matrix ${}^G\mathbf{p}_I \in \mathbb{R}^3$ and the rotation matrix ${}^G\mathbf{R}_I \in \text{SO}(3)$. The transformation of $\{C\}$ with respect to $\{I\}$ is expressed by the rotation matrix ${}^I\mathbf{R}_C$ and the translation matrix ${}^I\mathbf{p}_C$.

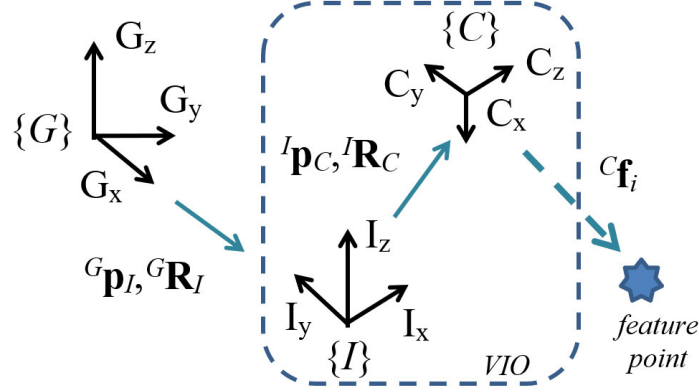


Figure 3.1: The VIO coordinate system

The rotation is operated with quaternion approach, satisfying the unit-length constraints (3.1). The quaternion approach shows the 2-1 covering homomorphism with the rotation group $\text{SO}(3)$ (i.e. double covering of the rotation group). However, this problem tends not to create any difficulty when employed for locally convergent filters. Quaternion representations are popularly applied in many navigation projects. $\mathbf{R}(\bar{\mathbf{q}}_1)$ is a function producing the rotational matrix from the quaternion $\bar{\mathbf{q}}_1$. The quaternion multiplication between $\bar{\mathbf{q}}_1 = [q_{w,1} \ q_{x,1} \ q_{y,1} \ q_{z,1}]^T$ and $\bar{\mathbf{q}}_2 = [q_{w,2} \ q_{x,2} \ q_{y,2} \ q_{z,2}]^T$ is computed as (3.2).

$$|\bar{\mathbf{q}}| = \sqrt{\bar{\mathbf{q}}^T \bar{\mathbf{q}}} = 1 \quad (3.1)$$

$$\bar{\mathbf{q}}_1 \otimes \bar{\mathbf{q}}_2 = \begin{bmatrix} q_{w,1}q_{w,2} - q_{x,1}q_{x,2} - q_{y,1}q_{y,2} - q_{z,1}q_{z,2} \\ q_{w,1}q_{x,2} + q_{x,1}q_{w,2} + q_{y,1}q_{z,2} - q_{z,1}q_{y,2} \\ q_{w,1}q_{y,2} - q_{x,1}q_{z,2} + q_{y,1}q_{w,2} + q_{z,1}q_{x,2} \\ q_{w,1}q_{z,2} + q_{x,1}q_{y,2} - q_{y,1}q_{x,2} + q_{z,1}q_{w,2} \end{bmatrix} \quad (3.2)$$

This rotation matrix has essential properties of special orthogonal group $\text{SO}(3)$ as $\mathbf{R} \in \text{SO}(3) \doteq \{\mathbf{R} \in \mathbb{R}^{3 \times 3} : \mathbf{R}^T \mathbf{R} = \mathbf{R} \mathbf{R}^T = \mathbf{I}, \det(\mathbf{R}) = 1\}$. $\mathfrak{so}(3)$ is denoted as the tangent space to the group $\text{SO}(3)$'s manifold (at the identity), and coincides with the space of skew symmetric matrices. The skew symmetric matrix with a vector in \mathbb{R}^3 is determined as in (5.1) following the property: $\forall \mathbf{a}, \mathbf{b} \in \mathbb{R}^{3 \times 3}, [\mathbf{a}]_{\times} \mathbf{b} = -[\mathbf{b}]_{\times} \mathbf{a}$.

$$S(\boldsymbol{\omega}) = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}^{\wedge} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \in \mathfrak{so}(3) \quad (3.3)$$

Special Euclidean Group $\text{SE}(3)$ is also mentioned for describing the group of 3D rigid motion, $\text{SE}(3) = \{(\mathbf{R}, \mathbf{p}) : \mathbf{R} \in \text{SO}(3), \mathbf{p} \in \mathbb{R}^3\}$. This group operations are listed as: $\mathbf{T}_1 \mathbf{T}_2 = (\mathbf{R}_1 \mathbf{R}_2, \mathbf{R}_1 \mathbf{p}_2 + \mathbf{p}_1)$ and $\mathbf{T}_1^{-1} = (\mathbf{R}_1^T, -\mathbf{R}_1^T \mathbf{p}_1)$.

3.2.2 IMU Sensor Model

At time k , the IMU sensor provides accelerometer and gyroscope measurements (${}^I \mathbf{a}_m$ and ${}^I \boldsymbol{\omega}_m$), which are expressed in three directions (x , y and z) with metric units in $\{I\}$ frame. Those measurements can be modeled as in (3.4).

$$\begin{aligned} {}^I \mathbf{a}_m(k) &= \mathbf{R}({}^I \bar{\mathbf{q}}_G(k)) ({}^G \mathbf{a}_I(k) - {}^G \mathbf{g}) + \mathbf{b}_a(k) + \mathbf{n}_a(k) \\ {}^I \boldsymbol{\omega}_m(k) &= {}^I \boldsymbol{\omega}(k) + \mathbf{b}_g(k) + \mathbf{n}_g(k) \end{aligned} \quad (3.4)$$

where ${}^G\mathbf{a}_I$ is the linear acceleration of the IMU with respect to the frame $\{G\}$; ${}^G\mathbf{g}$ denotes the gravitational acceleration in the frame $\{G\}$; ${}^I\boldsymbol{\omega}$ is the angular velocity of the IMU in $\{I\}$ frame. \mathbf{b}_a and \mathbf{b}_g are 3×1 vectors representing the slowly-varying bias of the accelerometer and gyroscope. The residual noise \mathbf{n}_a and \mathbf{n}_g are modeled as white Gaussian noise.

3.3 IMU Filter State Propagation

3.3.1 Filter State Formation

Similar to the common approach [64, 79], IMU data are used for filter state prediction. The true state of IMU at the time k can be defined as $\mathbf{x}_{\text{IMU},k} = \begin{bmatrix} {}^G\mathbf{p}_I^T & {}^G\mathbf{q}_I^T & {}^G\mathbf{v}_I^T & \mathbf{b}_a^T & \mathbf{b}_g^T \end{bmatrix}^T$, which accords with the kinematic equations (3.5). ${}^G\mathbf{p}_I$ and ${}^G\mathbf{v}_I$ denote the IMU position and velocity with respect to the frame $\{G\}$, respectively. \mathbf{n}_{wa} and \mathbf{n}_{wg} are 3×1 white Gaussian noise vectors.

$$\begin{aligned} {}^G\dot{\mathbf{v}}_I &= {}^G\mathbf{a}_I \\ {}^G\dot{\mathbf{q}}_I &= \frac{1}{2} {}^G\bar{\mathbf{q}}_I \otimes \begin{bmatrix} 0 & {}^I\boldsymbol{\omega}^T \end{bmatrix}^T; \quad {}^G\dot{\mathbf{p}}_I = {}^G\mathbf{v}_I \\ \dot{\mathbf{b}}_a &= \mathbf{n}_{wa}; \quad \dot{\mathbf{b}}_g = \mathbf{n}_{wg} \end{aligned} \quad (3.5)$$

The IMU true state is expressed by a combination of the nominal state $\hat{\mathbf{x}}_{\text{IMU},k}$ (3.6) and the error state $\tilde{\mathbf{x}}_{\text{IMU},k}$ (3.7). The function $\mathbf{g}(\hat{\mathbf{x}}_{\text{IMU},k}, \tilde{\mathbf{x}}_{\text{IMU},k})$ (3.8) is used for describing that combination:

$$\hat{\mathbf{x}}_{\text{IMU},k} = \begin{bmatrix} {}^G\hat{\mathbf{p}}_I^T & {}^G\hat{\mathbf{q}}_I^T & {}^G\hat{\mathbf{v}}_I^T & \hat{\mathbf{b}}_a^T & \hat{\mathbf{b}}_g^T \end{bmatrix}^T \quad (3.6)$$

$$\tilde{\mathbf{x}}_{\text{IMU},k} = \begin{bmatrix} {}^G\tilde{\mathbf{p}}_I^T & {}^G\delta\theta_I^T & {}^G\tilde{\mathbf{v}}_I^T & \tilde{\mathbf{b}}_a^T & \tilde{\mathbf{b}}_g^T \end{bmatrix}^T \quad (3.7)$$

$$\begin{aligned}
{}^G\bar{\mathbf{q}}_I &= {}^G\hat{\mathbf{q}}_I \otimes {}^G\delta\bar{\mathbf{q}}_I \\
{}^G\mathbf{p}_I &= {}^G\hat{\mathbf{p}}_I + {}^G\tilde{\mathbf{p}}_I; \quad {}^G\mathbf{v}_I = {}^G\hat{\mathbf{v}}_I + {}^G\tilde{\mathbf{v}}_I \\
\mathbf{b}_a &= \hat{\mathbf{b}}_a + \tilde{\mathbf{b}}_a; \quad \mathbf{b}_g = \hat{\mathbf{b}}_g + \tilde{\mathbf{b}}_g
\end{aligned} \tag{3.8}$$

Notably, the error state follows the standard additive error definition (i.e., arithmetic difference) when describing position, velocity and IMU biases. For rotation description, the error quaternion should not be expressed in terms of the arithmetic difference because of the presence of unit-length constraints (3.1). In fact, those constraints make the corresponding covariance matrix singular, which is difficult to maintain numerically during the quaternion computation [137]. For stability, a different representation is used for the rotational error state vector, ${}^G\delta\theta_I$. When combining with the nominal state (3.8), the error quaternion is constructed from the rotational error as ${}^G\delta\bar{\mathbf{q}}_I = \begin{bmatrix} 1 & \frac{1}{2} {}^G\delta\theta_I^T \end{bmatrix}^T$.

The VIO system is designed using the structure of MSCKF, in which the filter state includes the current IMU state and multiple previous camera poses. In this chapter, the point transfer approach using trifocal tensor geometry is applied for the measurement update step. This approach only requires two previous camera poses in the filter state. These camera poses are associated with two reference images, I_1 and I_2 . Additionally, the state is modified to include the previous IMU poses (${}^G\mathbf{p}_{I_1}$, ${}^G\bar{\mathbf{q}}_{I_1}$; ${}^G\mathbf{p}_{I_2}$, ${}^G\bar{\mathbf{q}}_{I_2}$) instead of the camera poses for advantageous implementation and state transition. The transformation between them is easily obtained by using the static IMU-camera calibrated transformation.

3.3.2 Filter Propagation

For discrete-time implementation of the filter state propagation, IMU measurements (\mathbf{a}_m and ω_m) are sampled with a period Δt . Those measurements will be used for propagat-

ing the filter state $\hat{\mathbf{x}}_{k|k-1}$ using (3.5) and 4th order Runge-Kutta numerical integration [95]. According to the researches [64, 147], the filter error state will operate following the kinematic equations (3.9). Those equations can be rewritten in a form that corresponds to the linearized continuous-time error state equation, as in (3.10).

$$\begin{aligned} {}^G\dot{\tilde{\mathbf{p}}}_I &= {}^G\tilde{\mathbf{v}}_I \\ {}^G\delta\dot{\theta}_I &= -S({}^I\boldsymbol{\omega}_m - \hat{\mathbf{b}}_g) {}^G\delta\theta_I - \tilde{\mathbf{b}}_g - \mathbf{n}_g \end{aligned} \quad (3.9)$$

$${}^G\dot{\tilde{\mathbf{v}}}_I = -\mathbf{R}({}^G\hat{\mathbf{q}}_I) S({}^I\mathbf{a}_m - \hat{\mathbf{b}}_a) - \mathbf{R}({}^G\hat{\mathbf{q}}_I)\tilde{\mathbf{b}}_a - \mathbf{R}({}^G\hat{\mathbf{q}}_I)\mathbf{n}_a$$

$$\dot{\tilde{\mathbf{b}}}_a = \mathbf{n}_{wa}; \quad \dot{\tilde{\mathbf{b}}}_g = \mathbf{n}_{wg}$$

$$\dot{\tilde{\mathbf{x}}}_k = \mathbf{F}_c\tilde{\mathbf{x}}_k + \mathbf{G}_c\mathbf{n}_{\text{IMU}} \quad (3.10)$$

where $\mathbf{n}_{\text{IMU}} = [\mathbf{n}_g, \mathbf{n}_a, \mathbf{n}_{wa}, \mathbf{n}_{wg}]^T$ is the noise vector with variance $\sigma_g^2, \sigma_a^2, \sigma_{wa}^2, \sigma_{wg}^2$, respectively. \mathbf{F}_c is discretized to have \mathbf{F}_d as in (3.11). The discrete-time system noise covariance matrix \mathbf{Q}_d can be constructed from the continuous-time system noise covariance matrix $\mathbf{Q}_c = \text{diag}(\sigma_g^2, \sigma_a^2, \sigma_{wa}^2, \sigma_{wg}^2)$ as in (3.12). The values of \mathbf{Q}_c can be defined through offline IMU sensor calibration. Finally, the state covariance matrix $\mathbf{P}_{k|k-1}$ can be propagated by \mathbf{F}_d and the error process noise covariance matrices \mathbf{Q}_d , as in (3.13). More information about the matrix structure of \mathbf{F}_c , \mathbf{F}_d , \mathbf{Q}_c and \mathbf{Q}_d can be found in [137, 147].

$$\mathbf{F}_d = \exp(\mathbf{F}_c\Delta t) = \mathbf{I}_d + \mathbf{F}_c\Delta t + \frac{1}{2!}\mathbf{F}_c^2\Delta t^2 + \dots \quad (3.11)$$

$$\mathbf{Q}_d = \int_{\Delta t} \mathbf{F}_d(\tau) \mathbf{G}_c \mathbf{Q}_c \mathbf{G}_c^T \mathbf{F}_d^T(\tau) d\tau \quad (3.12)$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_d \mathbf{P}_{k-1|k-1} \mathbf{F}_d^T + \mathbf{Q}_d \quad (3.13)$$

3.3.3 Filter Measurement Update with Cubature Rule

The cubature Kalman filter is a Jacobian-free nonlinear filter, which applies a deterministic sampling approach to handle the nonlinearity of the TTG-based visual measurement model. CKF shares common properties with the unscented Kalman filter but is improved in numerical implementation and system stability. The authors [18] have addressed the vulnerability of the UKF system failure associated with the negatively-weighted sigma points. The presence of those points can halt the filter operation or even cause the system to fail when taking the square-root operation of the covariance matrix. That vulnerable step is improved in CKF by using Spherical-Radial Transformation (**SRT**) and cubature points.

CKF handles the nonlinearity by generating a set of cubature points following the spherical-radial cubature rule. Some overview of CKF is provided as follows. \mathbf{D}_k is denoted as the history of measurement pairs up to time step k , while $\mathcal{N}(\cdot, \cdot)$ is the conventional symbol for a Gaussian density. In order to develop the Bayesian filter theory under Gaussian domain, CKF assumes that the predictive density $p(\mathbf{x}_k | \mathbf{D}_{k-1})$ and the filter likelihood density $p(\mathbf{z}_k | \mathbf{D}_k)$ are both Gaussian, which eventually leads to a Gaussian posterior density $p(\mathbf{x}_k | \mathbf{D}_k)$. Under that Gaussian-approximation assumption, the functional recursion of the Bayesian filter is reduced to the estimate of mean and covariance of various conditional densities, following the time and the measurement update [18]. Both prediction and update equations have the form of computing the multi-dimensional integrals, which consist of the *non-linear function* \times *Gaussian* such as (3.14). Those integrals are solved using cubature rules. A random variable \mathbf{x} has mean $\hat{\mathbf{x}}$ and covariance \mathbf{P} . Similar to the Unscented Transform of UKF, SRT is a method to compute the statistics of a random variable \mathbf{x} which undergoes a nonlinear transformation, $\mathbf{y} = f(\mathbf{x})$ as in Fig.3.2. A new set of cubature points

($s = 1, 2, \dots, 2n$) around the mean $\hat{\mathbf{x}}$ is generated to capture a number of low-order moments of the prior density $p(\mathbf{x})$ with the Gaussian assumption for the prior statistics of \mathbf{x} . Those cubature points are propagated through the nonlinear transformation, $\mathbf{Y}_s = \mathbf{f}(\mathbf{X}_s)$. The mean and covariance of \mathbf{y} are calculated approximately by using a weighted sample mean \mathbf{Y}_s and the covariance of the posterior cubature points.

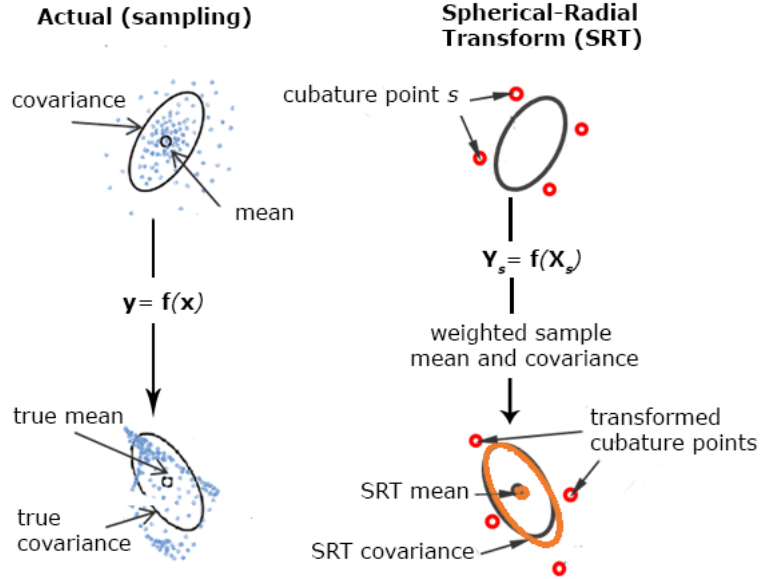


Figure 3.2: Illustration of the spherical-radial transformation using in CKF [18]

CKF is a straightforward implementation of SRT in the recursive estimation. If a system has n state variables, the third-order CKF selects $2n$ cubature points in order to compute the standard Gaussian weighted integral:

$$\begin{aligned}\hat{\mathbf{x}}_{k|k-1} &= \int_{\mathbb{R}^n} \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) \mathcal{N}(\mathbf{x}_{k-1}; \hat{\mathbf{x}}_{k-1|k-1}, \mathbf{P}_{k-1|k-1}) d\mathbf{x}_{k-1} \\ \hat{\mathbf{z}}_{k|k-1} &= \int_{\mathbb{R}^n} \mathbf{h}(\mathbf{x}_k) \mathcal{N}(\mathbf{x}_{k-1}; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}) d\mathbf{x}_k\end{aligned}\tag{3.14}$$

If a system has n state variables, the third-order cubature Kalman filter selects $2n$ cuba-

ture points in order to compute the standard Gaussian integral [18, 108]:

$$I_{\mathcal{N}}(\mathbf{f}) = \int_{\mathbb{R}^n} \mathbf{f}(\mathbf{x}) \mathcal{N}(\mathbf{x}; \hat{\mathbf{x}}, \mathbf{P}) d\mathbf{x} \simeq \sum_{s=1}^{2n} \frac{1}{2n} \mathbf{f}(\hat{\mathbf{x}} + \mathbf{S}\xi_s) \quad (3.15)$$

where the square-root factor \mathbf{S} of the covariance \mathbf{P} satisfies the equality $\mathbf{P} = \mathbf{S}\mathbf{S}^T$. The cubature points ξ_s are given by (3.16).

$$\xi_s = \begin{cases} \sqrt{n}e_s & s = 1, 2, \dots, n \\ -\sqrt{n}e_{s-n} & s = n+1, n+2, \dots, 2n \end{cases} \quad (3.16)$$

where $e_s \in \mathbb{R}^{n \times 1}$ represents the s^{th} elementary column vector.

Using those cubature points, the measurement update step is performed to correct the predicted state $\hat{\mathbf{x}}_{k|k-1}$ and the associated covariance $\mathbf{P}_{k|k-1}$. Firstly, the predicted covariance matrix $\mathbf{P}_{k|k-1}$ is factorized by (3.17).

$$\mathbf{P}_{k|k-1} = \mathbf{S}_{k|k-1} \mathbf{S}_{k|k-1}^T \quad (3.17)$$

Next, a new set of cubature points ($s = 1, 2, \dots, 2n$) is computed from the predicted state $\hat{\mathbf{x}}_{k|k-1}$ and the square-root factor $\mathbf{S}_{k|k-1}$ by (3.18). Those cubature points are then evaluated using with the nonlinear measurement model, $\mathbf{h}(\cdot)$, to obtain the sampled measurement points (3.19). The innovation covariance matrix \mathbf{P}_{zz} and the cross covariance matrix \mathbf{P}_{xz} are computed in (3.20) and (3.21) prior to the Kalman gain calculation. The Kalman gain \mathbf{K}_k is computed with the visual measurement noise \mathbf{R} in (3.22). Then, the measurement residual $\tilde{\mathbf{z}}_k$, the posterior state and the associated covariance matrix are corrected with the actual measurement \mathbf{z}_k from (3.23) and (3.24), respectively.

$$\mathbf{X}_{s,k|k-1} = \mathbf{S}_{k|k-1} \xi_s + \hat{\mathbf{x}}_{k|k-1} \quad (3.18)$$

$$\begin{aligned}\mathbf{Z}_{s,k|k-1} &= \mathbf{h}(\mathbf{X}_{s,k|k-1}, \{\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3\}) \\ \hat{\mathbf{z}}_{k|k-1} &= \sum_{s=1}^{2n} \frac{1}{2n} \mathbf{Z}_{s,k|k-1}\end{aligned}\tag{3.19}$$

$$\mathbf{P}_{xz} = \sum_{s=1}^{2n} \frac{1}{2n} (\mathbf{X}_{s,k|k-1} - \hat{\mathbf{x}}_{k|k-1}) (\mathbf{Z}_{s,k|k-1} - \hat{\mathbf{z}}_{k|k-1})^T \tag{3.20}$$

$$\mathbf{P}_{zz} = \sum_{s=1}^{2n} \frac{1}{2n} (\mathbf{Z}_{s,k|k-1} - \hat{\mathbf{z}}_{s,k|k-1}) (\mathbf{Z}_{s,k|k-1} - \hat{\mathbf{z}}_{s,k|k-1})^T \tag{3.21}$$

$$\mathbf{K}_k = \mathbf{P}_{xz} (\mathbf{P}_{zz} + \mathbf{R})^{-1} \tag{3.22}$$

$$\tilde{\mathbf{z}}_k = \mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1}, \quad \hat{\mathbf{x}}_{k|k} = \mathbf{g}(\hat{\mathbf{x}}_{k|k-1}, \mathbf{K}_k \tilde{\mathbf{z}}_k) \tag{3.23}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{P}_{zz} \mathbf{K}_k^T \tag{3.24}$$

Equations 3.20 and 3.21 are presented in the generic form of CKF algorithm. Meanwhile, the VIO prediction design defines the nominal state $\tilde{\mathbf{x}}_{30 \times 1}$ and the error state $\tilde{\mathbf{x}}_{27 \times 1}$ with different sizes because of the quaternion representation. Hence, 3.20 and 3.21 should be adapted for implementation using the function (3.8). After the measurement update step is performed, the filter needs to perform state transition. $(I_1, {}^G\mathbf{p}_{I_1}, {}^G\bar{\mathbf{q}}_{I_1})$ is replaced by $(I_2, {}^G\mathbf{p}_{I_2}, {}^G\bar{\mathbf{q}}_{I_2})$ and $(I_2, {}^G\mathbf{p}_{I_2}, {}^G\bar{\mathbf{q}}_{I_2})$ is replaced by $(I_3, {}^G\mathbf{p}_I, {}^G\bar{\mathbf{q}}_I)$. The covariance matrix is also updated in that order.

Figure 3.3 presents the system architecture of the proposed VIO algorithm. The *Image Processing* block is assigned for feature detection and matching. Visual landmark features are detected in images I_1 , I_2 and I_3 . Multiple techniques for feature detection can be used such as the Speed-Up Robust Feature (**SURF**) [21] and Scale-Invariant Feature Transform (**SIFT**) [88], which have the properties of scale-invariance and rotation-invariance. In addition, the feature detection is also adaptive with the acceptable change of ambient lighting condition, the viewpoint disparity and image noise. In this chapter, the SIFT feature is used

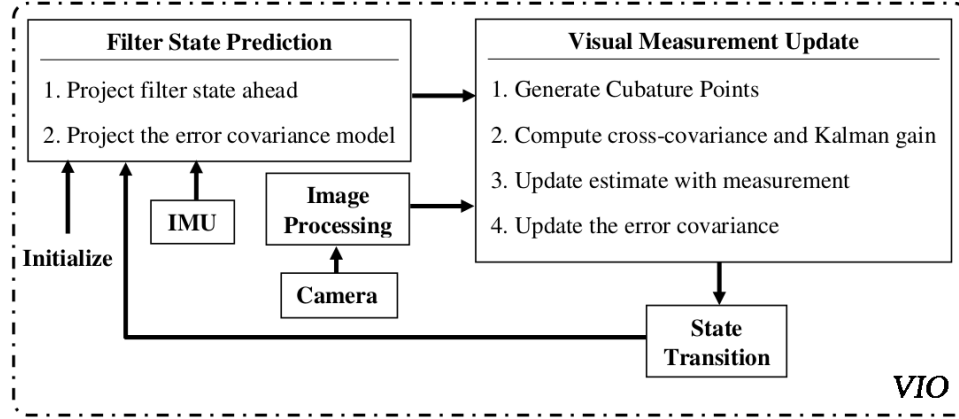


Figure 3.3: System architecture of the proposed algorithm

for feature detection. These detected feature points in each image are matched together. Due to image noise, image blur, occlusion and other environmental elements, some mismatched features are unavoidable. Hence, 1-Point Random Sample Consensus (**RANSAC**) technique [35, 36, 64] is applied to eliminate the outliers or mismatched features. The 1-Point RANSAC technique is an improvement on traditional RANSAC [51], which employs camera pose information to help eliminate mismatched feature points.

3.4 Trifocal Tensor Geometry Based Measurement Model

Most of the existing VIO systems attempt to reconstruct the 3D feature points before predicting the measurements [79, 95]. In the context of Structure from Motion (**SFM**), many approaches have been introduced to recover the 3D feature points from the camera images such as two frames SFM using fundamental matrix [60], an inverse-depth least-squares Gauss-Newton optimization [94], factorization with multi-frame SFM [135], and bundle adjustment [140]. However, VIO implementation needs to achieve the sufficient frequency

update rate to keep track of the vehicle under varying conditions of observation. For this purpose, the inverse-depth least-squares Gauss-Newton optimization approach is selected to perform the reconstruction [94, 95]. This approach can achieve the accurate reconstruction accuracy and compromise the computational cost.

Each feature is tracked through n consecutive images (Fig.3.4), which improves the optimization accuracy but increases the computational cost. However, it is feasible to predict the measurements without defining the point in space or reconstructing the 3D geometry explicitly. This chapter utilizes the relation between measurements in the images. The point transfer using Trifocal Tensor Geometry, which only uses three consecutive images ($n = 3$), is applied to compute the predicted measurements. This section describes an analysis of the TTG-based approach and compares it to the 3D feature-point approach.

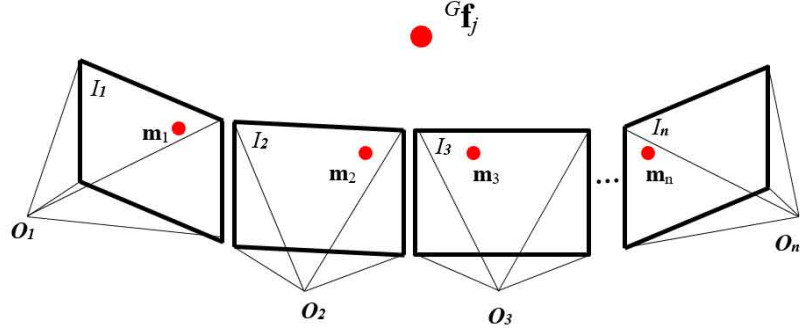


Figure 3.4: Illustration of the 3D feature-point reconstruction

3.4.1 Point Transfer Using Trifocal Tensor Geometry

u, v are denoted as pixel coordinates of the 2D feature point \mathbf{m} . \mathbf{m} is presented in a homogeneous coordinate with $\tilde{\mathbf{m}} = [u, v, 1]^T$. The point transfer approach using TTG is summarized by the followers. According to [60], the camera projection matrices for the three different

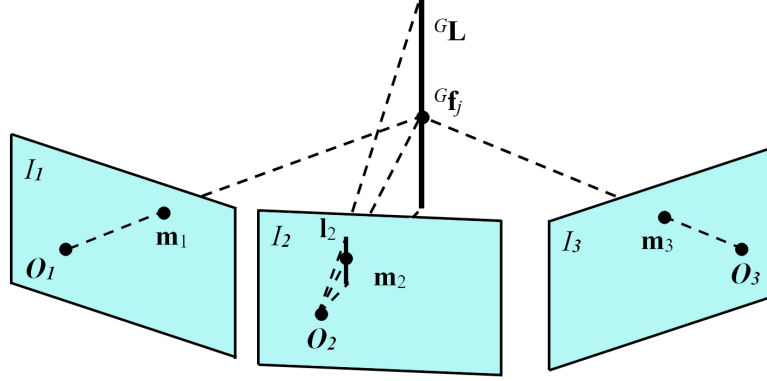


Figure 3.5: Trifocal tensor incidence relation (point-line-point) for three views I_1 , I_2 , and I_3 with camera viewpoints O_1 , O_2 , and O_3

views are described as in (3.25). The point-line-point correspondence ($\tilde{\mathbf{m}}_1 \leftrightarrow \mathbf{l}_2 \leftrightarrow \tilde{\mathbf{m}}_3$) is established with three image views I_1 , I_2 and I_3 , indicated by their viewpoints O_1 , O_2 , and O_3 (Fig. 3.5).

$$\begin{aligned}\mathbf{P}_1 &= [\mathbf{I}_{3 \times 3} | \mathbf{0}_{3 \times 1}] \\ \mathbf{P}_2 &= [\mathbf{A} | \mathbf{a}_4] = [\mathbf{R}_{12}^T | -\mathbf{R}_{12}^T \mathbf{t}_{12}] \\ \mathbf{P}_3 &= [\mathbf{B} | \mathbf{b}_4] = [\mathbf{R}_{13}^T | -\mathbf{R}_{13}^T \mathbf{t}_{13}]\end{aligned}\tag{3.25}$$

$$\begin{aligned}{}^G\mathbf{R}_{C_1} &= \mathbf{R}({}^G\bar{\mathbf{q}}_{I_1})^T \mathbf{R}_C; & {}^G\mathbf{p}_{C_1} &= {}^G\mathbf{p}_{I_1} + \mathbf{R}({}^G\bar{\mathbf{q}}_{I_1})^T {}^G\mathbf{p}_C \\ {}^G\mathbf{R}_{C_2} &= \mathbf{R}({}^G\bar{\mathbf{q}}_{I_2})^T \mathbf{R}_C; & {}^G\mathbf{p}_{C_2} &= {}^G\mathbf{p}_{I_2} + \mathbf{R}({}^G\bar{\mathbf{q}}_{I_2})^T {}^G\mathbf{p}_C \\ \mathbf{R}_{12} &= ({}^G\mathbf{R}_{C_1})^T {}^G\mathbf{R}_{C_2}; & \mathbf{t}_{12} &= ({}^G\mathbf{R}_{C_1})^T ({}^G\mathbf{p}_{C_2} - {}^G\mathbf{p}_{C_1})\end{aligned}\tag{3.26}$$

where $(\mathbf{R}_{12}, \mathbf{t}_{12})$, $(\mathbf{R}_{13}, \mathbf{t}_{13})$, $(\mathbf{R}_{23}, \mathbf{t}_{23})$ are the transformations of three camera poses, which can be computed directly from the filter state (3.26). \mathbf{a}_i and \mathbf{b}_i are the i -th columns of the matrices \mathbf{P}_2 and \mathbf{P}_3 as well as the two epipoles \mathbf{a}_4 and \mathbf{b}_4 in the second and third views respectively, arising from the first camera. A trifocal tensor can be formulated as (3.27):

$$\mathbf{T}_i = \mathbf{a}_i \mathbf{b}_4^T - \mathbf{a}_4 \mathbf{b}_i^T\tag{3.27}$$

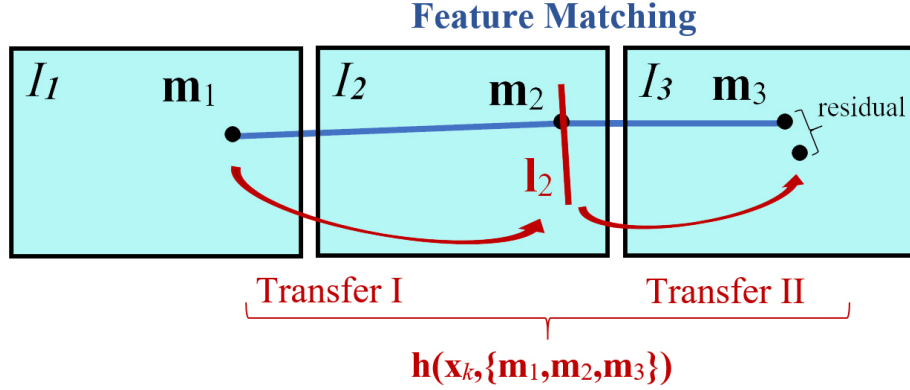


Figure 3.6: Illustration of the TTG based approach. The **red line** is the point transfer using TTG while the **blue line** is the feature-tracking pipeline

Having points \mathbf{m}_1 in I_1 , \mathbf{m}_2 in I_2 , and essential camera poses, it is able to predict point $\hat{\mathbf{m}}_3$ in I_3 using that tensor and 2 transfers (Fig. 3.6). In transfer I, the approach attempts to construct a line in the image I_2 passing the point \mathbf{m}_2 . The research [60] recommends selecting a line, which is perpendicular to the epipolar line \mathbf{l}_e : $\mathbf{l}_2 = [l_2, -l_1, -u_2l_2 + v_2l_1]^T$ with $\mathbf{l}_e = \mathbf{R}_{12}^T [\mathbf{t}_{12} \times] \tilde{\mathbf{m}}_1 = [l_1, l_2, l_3]^T$. Then, transfer II uses \mathbf{l}_2 and camera matrix $\mathbf{K}_{\text{camera}}$ for predicting $\hat{\mathbf{m}}_3$ as (3.28). Figure 3.7 demonstrates the point transfer using TTG with two transfers.

$$\hat{\mathbf{m}}_3 = \mathbf{K}_{\text{camera}} \left(\sum_i \tilde{\mathbf{m}}_{1i} \mathbf{T}_i^T \right) \mathbf{l}_2 \quad (3.28)$$

To apply TTG based measurement model to the proposed CKF, (3.29) is used as the $\mathbf{h}(\cdot)$ function of (3.19). This function will predict the feature-point measurement in the current image and guarantee the epipolar constraint of two images. Having the predicted measurement $\hat{\mathbf{z}}$ and actual measurement \mathbf{z} , the residual $\tilde{\mathbf{z}}$ and Kalman gain \mathbf{K} can be calculated to

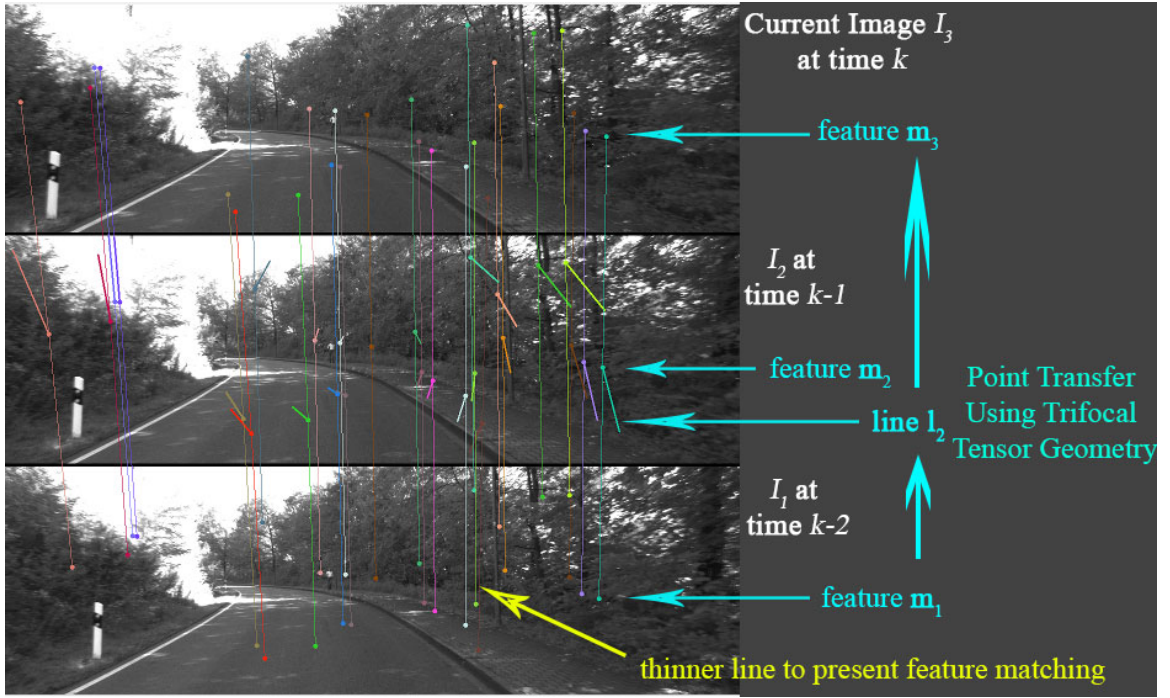


Figure 3.7: Example of point transfer using TTG. Images are obtained from KITTI dataset [56]. The point-line-point relation, which is represented with different colours, is established between the feature \mathbf{m}_1 in I_1 , the line \mathbf{l}_2 (thicker line) in I_2 and the feature \mathbf{m}_3 in I_3 . The feature matching is presented with the thinner line connecting three features of three images.

correct the predicted filter state.

$$\mathbf{h}(\mathbf{x}_k, \{\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3\}) = \begin{bmatrix} \mathbf{K}_{\text{camera}} \left(\sum_i \tilde{\mathbf{m}}_{1,i} \mathbf{T}_i^T \right) \mathbf{l}_2 \\ \tilde{\mathbf{m}}_2^T \mathbf{R}_{12}^T [\mathbf{t}_{12}]_{\times} \tilde{\mathbf{m}}_1 \\ \tilde{\mathbf{m}}_3^T \mathbf{R}_{23}^T [\mathbf{t}_{23}]_{\times} \tilde{\mathbf{m}}_2 \\ \tilde{\mathbf{m}}_3^T \mathbf{R}_{13}^T [\mathbf{t}_{13}]_{\times} \tilde{\mathbf{m}}_1 \end{bmatrix} \quad (3.29)$$

A mechanism to select a new camera image as a keyframe is applied using two thresholds. Since the overlap region should be sufficient to track many feature points, a certain threshold is utilized for the number of tracked features to declare a new keyframe. Additionally, if there are no motion between three cameras ($\mathbf{P}_1 = \mathbf{P}_2 = \mathbf{P}_3 = [\mathbf{I}_{3 \times 3} | \mathbf{0}_{3 \times 1}]$), $\hat{\mathbf{m}}_3$ becomes zero for any landmark. In this case, the approach is unable to predict the measurement. Hence, the ego-motion between two camera poses should be so obvious that the baseline is not too short. For this purpose, a threshold is set for an average parallax between the previous keyframe and the new camera image.

3.4.2 Comparison between TTG-based Approach and 3D Feature-point Approach

Figure 3.8 describes the structural difference between the two approaches in computing predicted visual measurements. Both approaches use the same procedure for feature tracking through n camera images. The TTG-based approach operates as a straightforward and non-recursive function. $n > 3$ can be set to improve the quality of feature tracking even though the TTG-based approach only uses three camera images. Experiments are performed with KITTI dataset and measure processing time to evaluate the benefit of the TTG-based measurement approach. For clarification, this approach attempts to accelerate

the computation of the predicted measurements in the Kalman filter. The benefit does not include the acceleration of feature detection and matching.

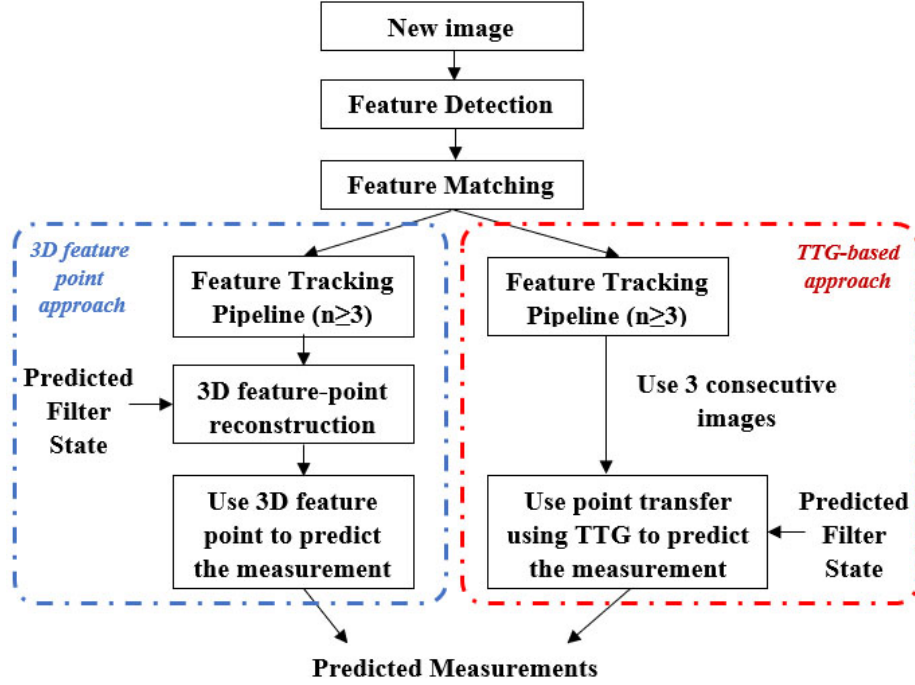


Figure 3.8: Structure of TTG-based approach and 3D feature-point approach

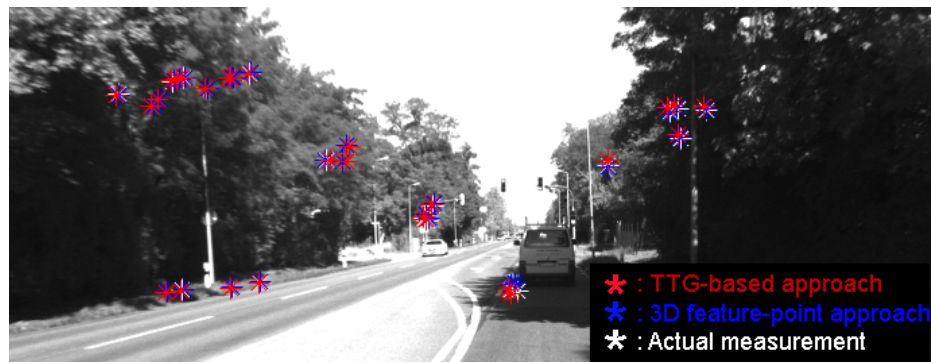


Figure 3.9: The results of the predicted measurements when verifying two approaches with KITTI dataset [56]

This subsection compares the result of the predicted measurements using two approaches when providing the identical inputs (i.e., feature points and predicted filter state). In Fig.3.9, two approaches produce very close predictions of feature-points. In addition, the average computational time is measured for the comparison (Fig.3.10) when implementing two models in Matlab. The TTG-based approach does not need to execute the least square Gauss-Newton optimization, which significantly decreases the computational time.

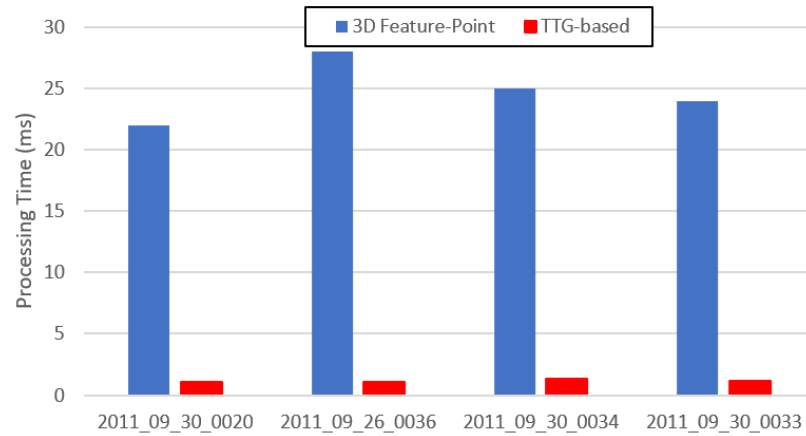


Figure 3.10: Average processing time to predict one feature in each approach

This chapter does not attempt to compare the two approaches in terms of the measurement residual, which does not effect directly to the filter estimation accuracy. In the Kalman filter, the measurement residual is used to correct the predicted filter state with a Kalman gain (3.23). The visual measurement noise \mathbf{R} determines the value of the Kalman gain and how the system weights the residual in the correction step. Hence, in the next section, the proposed VIO algorithm, which uses the TTG-based approach, is compared to the traditional VIO algorithm, which uses the 3D feature-point approach.

3.5 Experimental Results

3.5.1 Experiments with KITTI Dataset

The KITTI dataset [56] was collected by a ground vehicle in residential and rural areas. The VIO system includes a PointGray Flea2 camera with 1.4 Megapixels, 1392×512 pixel resolution, $90^\circ \times 35^\circ$ opening angle and a 10Hz update rate. An IMU sensor is an OXTS RT3003 6-axis L1L2 RTK with a resolution of 0.02m/0.1° and a 100Hz update rate. The dataset also provides accurate ground truth from a Velodyne laser scanner and a GPS localization system.

The KITTI dataset has many challenges for the VIO system operation. For example, Figs. 3.11.c and d were collected when the vehicle traveled in a forested environment on a sunny day. The trajectory was lined with trees on both sides. Consequently, this kind of environment made the lighting change quickly because of the sun position with respect to the camera and the number of tree canopies along the street. This dynamic lighting condition will challenge the feature tracking of the VIO system. Additionally, Figs.3.11.a, b and e illustrate the presence of unexpected moving objects in the camera images, such as cars and humans, which can affect the motion estimation negatively. The feature tracking with RANSAC based outlier rejection will eliminate such unreliable features of these moving objects from the visual measurement update step.

As shown in Fig.3.12, the proposed CKF algorithm can track the vehicle trajectory with satisfactory accuracy against the ground truth data. At each time instant, the vehicle velocity, bias of accelerometer and gyroscope are continuously updated, as in Figs.3.13, 3.14 and 3.15. The update rate from camera observation is sufficient to correct the IMU prediction as well as keep track of the vehicle trajectory.



Figure 3.11: Some camera images in KITTI dataset [56]

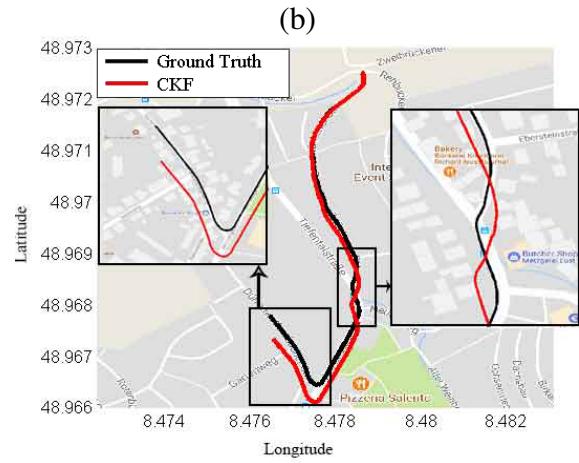
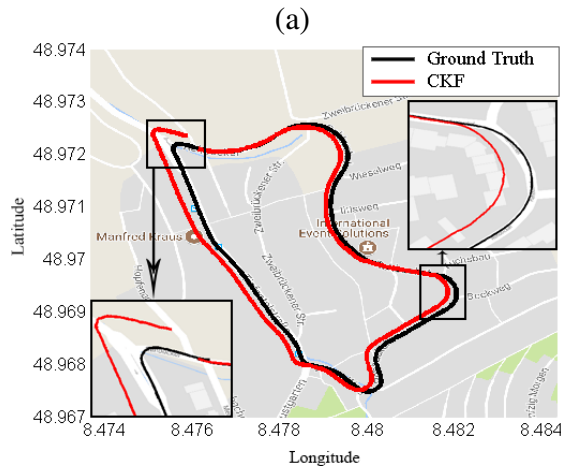
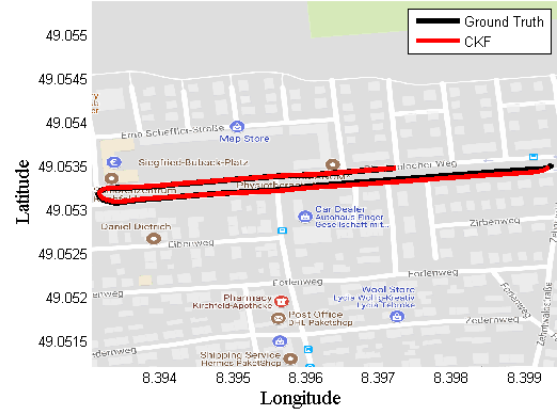
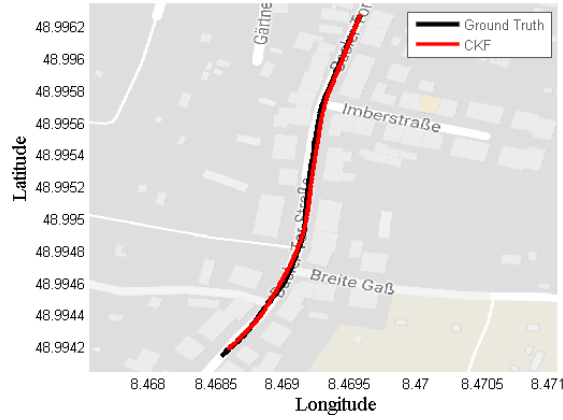


Figure 3.12: CKF estimation presented in Google map: (a) 2011_09_26_0095, (b) 2011_09_30_0020, (c) 2011_09_30_0033, (d) 2011_09_30_0034

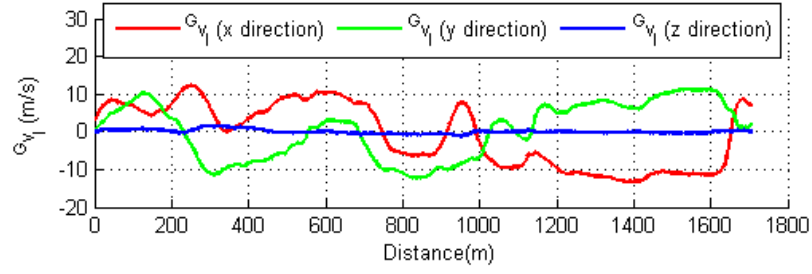


Figure 3.13: CKF estimation of velocity in dataset 2011_09_30_0034

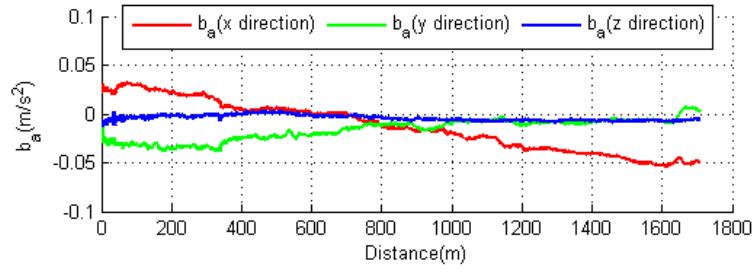


Figure 3.14: CKF estimation of accelerometer bias in dataset 2011_09_30_0034

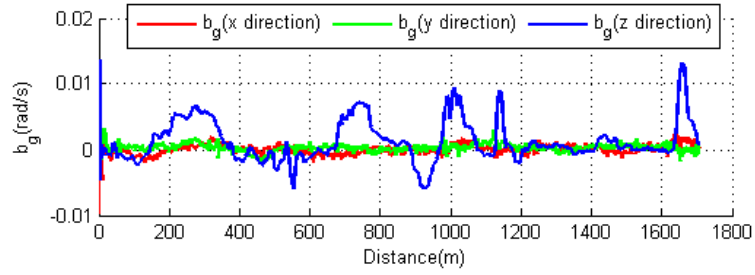


Figure 3.15: CKF estimation of gyroscope bias in dataset 2011_09_30_0034

It is useful to implement UKF [142] and compare it to CKF because both two filters share a common property of using a weighted set of symmetric points to handle the nonlinearity. Table 3.1 presents the experimental results of the proposed algorithm (CKF), UKF and EKF with Root Mean Square Error of position (**RMSE**) and rotation error (Ψ). The EKF implementation is the traditional MSCKF [79, 95]. The CKF and UKF use the TTG-based measurement model, while the EKF uses the 3D-feature point approach. The UKF implementation employs Cholesky factorization for the square-root operation on the covariance matrix. The UKF-computed covariance matrix has to be guaranteed to be positive definite. Otherwise, the negative sampling points may halt the filter operation or even cause the system to fail [18]. The CKF implementation has overcome these issues using cubature rule and singular value decomposition to factorize the covariance matrix. The CKF achieves better estimation accuracy than the UKF in VIO application. The CKF implementation has improved the system stability and the estimation accuracy relative to UKF. Further examples of CKF implementation can be found in [6]. The video demonstration of the CKF algorithm can be located at the following link: youtu.be/DuS7AKRu-7I.

The proposed algorithm is implemented with a DELL Inspiron 15 7000 computer having Intel® Core™ i7-7700HQ CPU @2.80GHz $\times 4$ and 16GB RAM. Figure 3.16 presents the average processing times of CKF and EKF execution. The CKF system with TTG does not perform the 3D feature-point reconstruction, which in turn reduces the processing time significantly. Although CKF and EKF implementations use different measurement models, it is interesting to make a comparison between CKF and EKF in terms of the Root Mean Square Error (**RMSE**) positional estimation. Firstly, the CKF's capability to handle the system nonlinearity helps to improve the estimation accuracy. Secondly, the 3D feature-point approach requires feature tracking through multiple camera images, classi-

Table 3.1: Experimental results for KITTI datasets with different travel distance [m]

| Dataset | Travel | CKF | | UKF | | EKF | |
|-----------------|----------|---------|------------------|---------|------------------|---------|------------------|
| | Distance | RMSE | $\Psi(^{\circ})$ | RMSE | $\Psi(^{\circ})$ | RMSE | $\Psi(^{\circ})$ |
| 2011_09_26_0046 | 46 | 0.2104 | 0.1856 | 0.4527 | 0.1807 | 0.2276 | 0.3210 |
| 2011_09_26_0095 | 247 | 1.6424 | 0.2228 | 1.6622 | 0.2105 | 1.8053 | 0.2415 |
| 2011_09_26_0036 | 738 | 4.3525 | 0.5289 | 5.2452 | 0.5387 | 4.8070 | 0.0185 |
| 2011_10_03_0047 | 743 | 5.9086 | 1.1535 | 10.5830 | 1.1317 | 24.9355 | 5.0962 |
| 2011_09_30_0020 | 766 | 8.4880 | 0.3298 | 12.9287 | 0.3364 | 19.9242 | 4.884 |
| 2011_09_30_0034 | 918 | 24.9519 | 0.5360 | 31.0495 | 0.5474 | × | × |
| 2011_09_30_0033 | 1721 | 20.5373 | 0.2322 | 16.6368 | 0.2377 | 51.295 | 0.2313 |

Note: EKF: Extended Kalman Filter; CKF: Cubature Kalman Filter; UKF: Unscented Kalman Filter; ×: the filter fails to track the trajectory.

fying reliable features into three groups: past, mature, and immature features [71]. Past and mature features are processed by the filter update step because they reach the defined maximum length of the sliding window. Meanwhile, the immature features are waiting to reach the maximum length to be processed. That procedure makes the filter use fewer measurements to correct the predicted filter state. The reduction may negatively affect the system robustness and accuracy, especially when operating under adverse conditions (e.g., areas with a limited number of features). On the other hand, the TTG-based approach only requires the feature tracks in the most recent three images of the feature-tracking pipeline. This makes the filter update step use more measurements to operate and achieve similar accuracy as the conventional approach. The experiment has also verified that the TTG-based approach can be utilized to replace the 3D feature-point approach to simplify the system in specific applications.

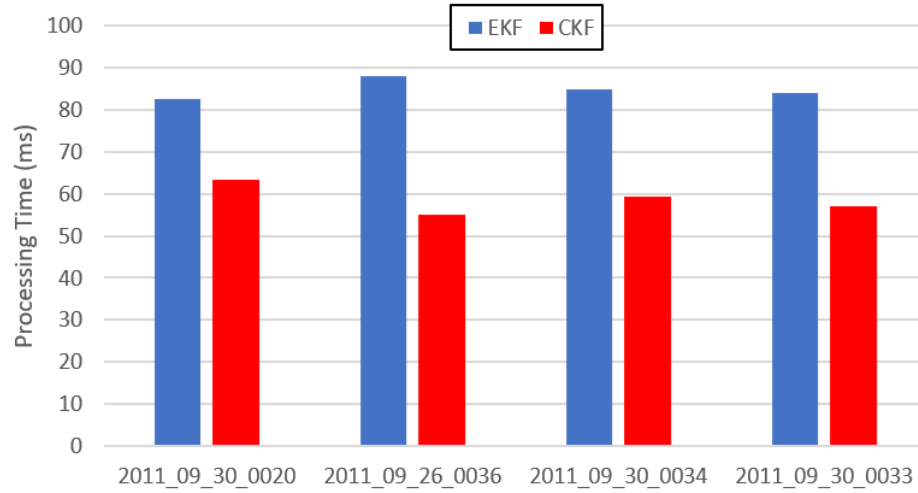


Figure 3.16: Processing time evaluation of CKF and EKF

The notable benefit of using the KITTI dataset is to verify the proposed algorithm per-

formance for long-distance travel. In table 3.1, the last dataset 2011_09_30.0033 is the case testing the filter estimation over long-term operation. All filters failed to track the vehicle trajectory over 1721m travel distance. In Fig.3.12, the error accumulation (drifts) can be observed in long-term operation. The accumulative error arises from different sources: (i) camera calibration, (ii) sensor resolution and (iii) the front-looking monocular camera with limited depth perception. The visual measurement update can not eliminate the accumulative errors effectively. Estimation over long-term operation also encounters the recovery issue when the camera frame is lost temporarily. The proposed algorithm does not generate the map during operation. These 3D landmarks are not available for the system recovery. If the camera observations (i.e., feature points) are recorded, the system can perform re-localization. The feature points observed in the current frame can be matched with those of the previous frames and estimate the relative transformation.

3.5.2 Experiments with EuRoC Dataset

The proposed algorithm is also verified with the EuRoC dataset [26], which is collected by a micro aerial vehicle in the indoor environment. The sensory platform of the EuRoC dataset operates at a higher rate than those of the KITTI dataset. The IMU signals are at 200Hz while the camera image is at 20Hz. The experiment only uses the left camera for the visual measurement update step. The maximum travel distance of each sequence in the EuRoC dataset is only about 100m.

Figure 3.17 attempts to compare the RMSE of the CKF's positional estimation against other state-of-the-art VIO algorithms such as VINS-Mono [119], ROVIO [23] and OKVIS [78] to determine the application scenario of the proposed algorithm. More rigorous com-

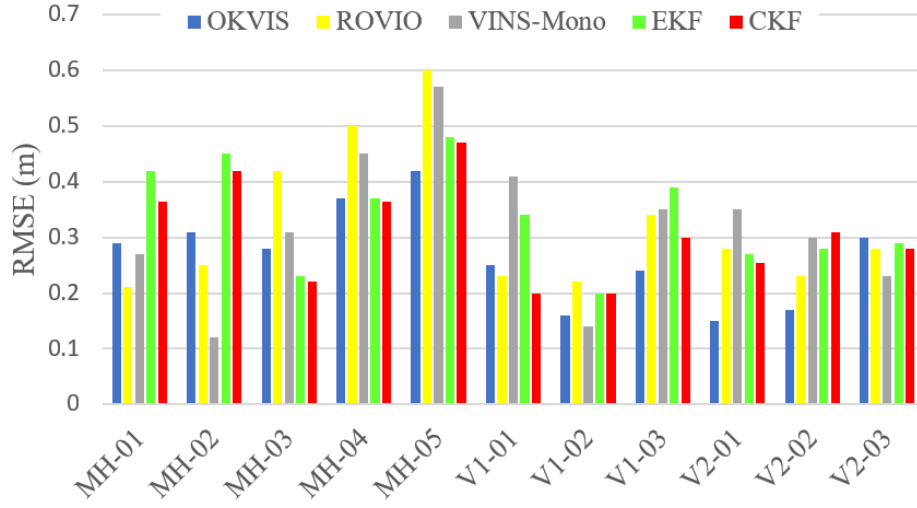


Figure 3.17: Experiments with EuRoC dataset [26]

parison and discussion of the MSCKF-based algorithm and other existing algorithms can be found in the research [44]. In these experiments, the proposed algorithm produces comparative estimation to other algorithms. VINS-Mono and OKVIS are classified as a smoothing technique, which formulates the estimation as a nonlinear optimization problem [53]. VINS-Mono performs full smoothing, which estimates for the entire history of camera poses and 3D landmarks. OKVIS limits the optimization within a sliding window as a fixed-lag smoothing. The optimization process minimizes the re-projection error of landmarks observed in camera frames to extract the trajectory and 3D landmarks. Consequently, the OKVIS can achieve better estimation accuracy than the filtering approach but require more computational resources to solve the optimization problem [44, 53, 90]. The proposed algorithm is classified as a filtering approach, which only focuses on inferring the current state directly from the sensor data. Compared to these smoothing techniques, our computation does not attempt to solve the optimization problem and generate the map of 3D points at each time instant, which requires fewer computational resources and processing

times. The implementation does not require the installation of an advanced optimization library such as Google's Ceres solver [16]. The proposed algorithm is beneficial to micro robotic systems having limited computational capability. Over a long-term operation, the proposed algorithm does not have an advanced function to effectively eliminate the error accumulation like other existing techniques such as VINS-Mono [119] and OKVIS [78]. Rather than deploying it as a stand-alone self-localization solution, the proposed algorithm is more useful for being part of larger navigation projects such as augmenting GPS.

3.5.3 Experiments with Handheld VIO Device

The proposed algorithm is also validated with a handheld VIO device. The device includes an Intel[®] RealSense[™] ZR300 VIO development kit [5] (Fig.3.18) and Intel[®] NUC mini computer. The IMU frequency is about 250Hz while the colour camera image frequency is about 20Hz. The mini computer is installed with Ubuntu 16.04, Robot Operation System (ROS)-Kinetic and the ROS driver of ZR300 module.

A dataset is collected in the Intelligent Systems Lab, EN1037, Engineering Building, Memorial University, shown in Fig.3.19. The ZR300 module is calibrated using the Kalibr visual-inertial calibration toolbox [122]. The calibration provides the intrinsic parameters of the colour camera, the IMU parameters and the transformation between the IMU frame and camera frame, which are necessary to the VIO implementation.

VINS-Mono [119] is also applied to extract the vehicle trajectory for comparison. VINS-Mono is an open-source VIO development, which implements a global-optimization based approach for sensor fusion. At each time instant, VINS-Mono attempts to optimize the entire trajectory based on the IMU data and multiple observations of the 3D feature

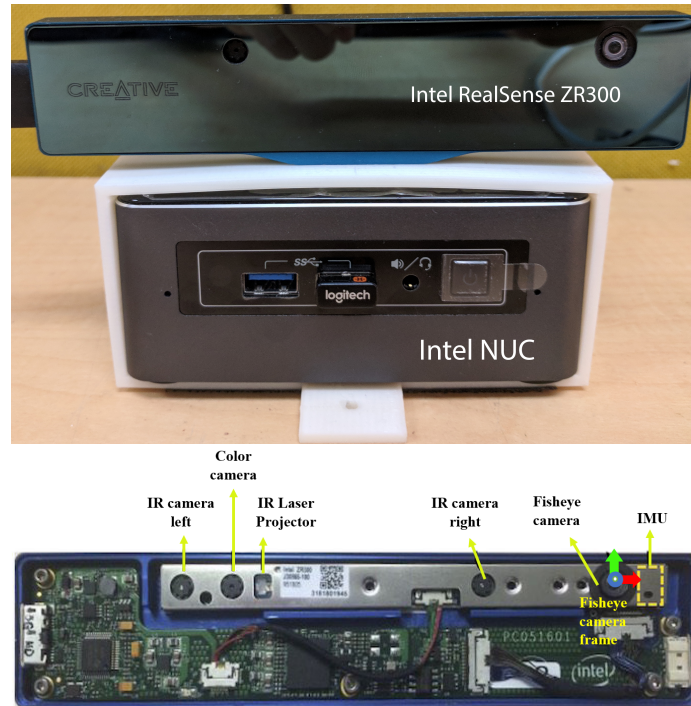


Figure 3.18: A handheld VIO device



Figure 3.19: Snapshots during the experiment with the handheld VIO device

points. When the vehicle makes a loop, VINS-Mono also activates the loop closure and re-localization functions to minimize the accumulative errors. The employment of these advanced functions results in the highest accuracy of the trajectory estimation, despite the considerable computational cost. In this experiment, the estimate of VINS-Mono is utilized as ground truth to evaluate the CKF estimation. Figures 3.20 and 3.21 shows the positional and rotational estimation of CKF. The CKF is able to track the vehicle trajectory.

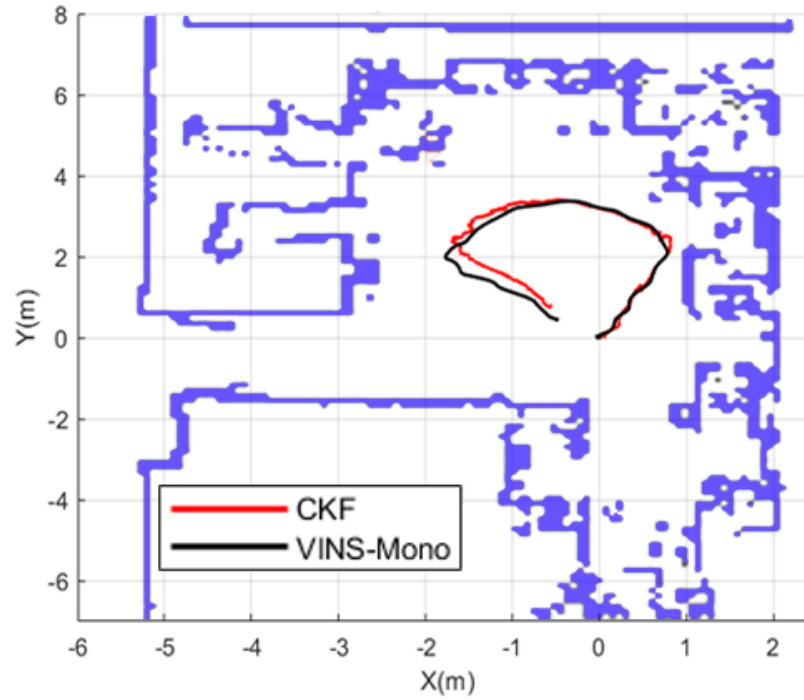


Figure 3.20: The trajectory of the handheld VIO device is estimated using the proposed algorithm and compared with the VINS-Mono estimate. The blue line is the map of the lab EN1037 generated by a laser scanner

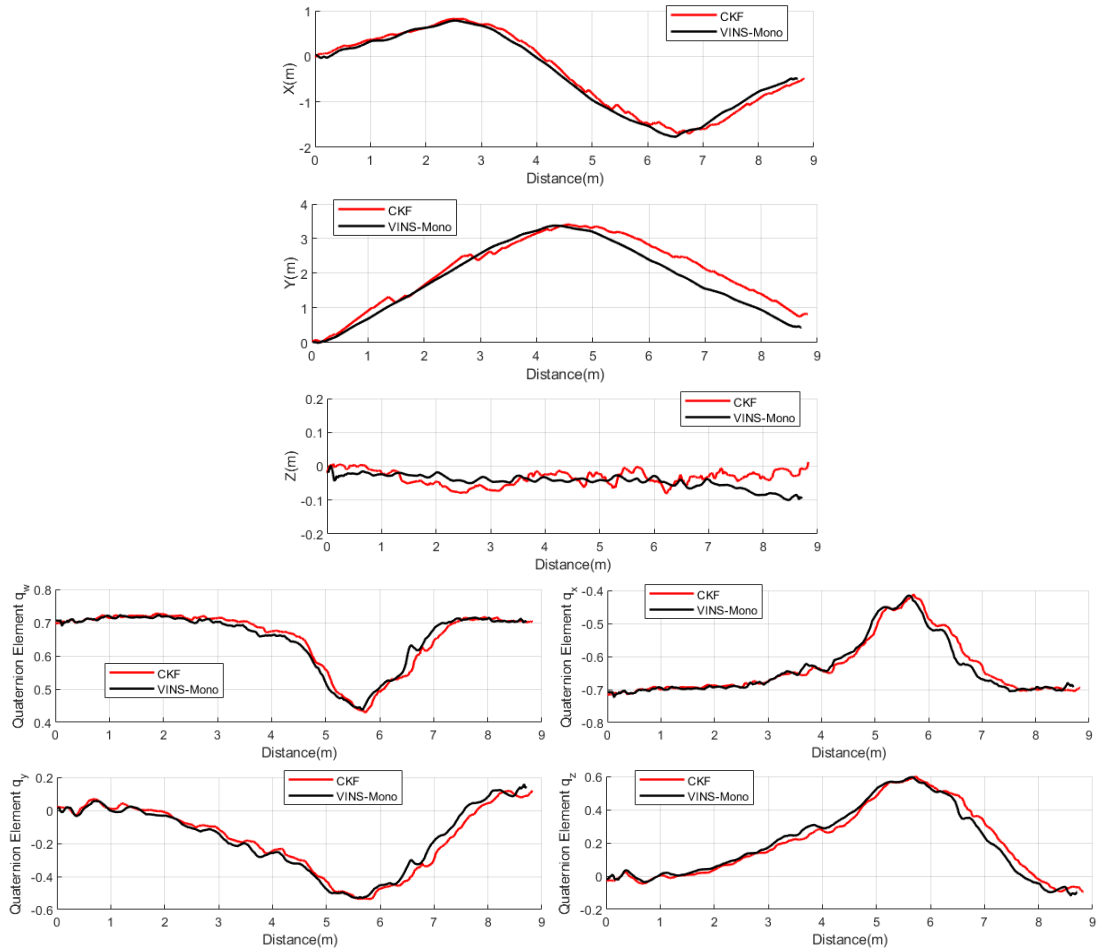


Figure 3.21: Positional and rotational estimations of the proposed algorithm and VINS-Mono

3.6 Cubature Information Filtering

3.6.1 Transformation to Information Domain

This chapter also applies an other variation of CKF known as the cubature information filter for VIO applications. The CIF addresses the computationally inefficient issue associated with Kalman structure. The CIF focuses on the information vector $\hat{\mathbf{y}}_{k|k}$ and information matrix $\mathbf{Y}_{k|k}$ instead of the state $\mathbf{x}_{k|k}$ and covariance matrix $\mathbf{P}_{k|k}$. The transformation is expressed as (3.30).

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{Y}_{k|k-1} \setminus \hat{\mathbf{y}}_{k|k-1}; \quad \mathbf{P}_{k|k-1} = \mathbf{Y}_{k|k-1} \setminus \mathbf{I} \quad (3.30)$$

The derivation of CIF is introduced in [108] and repeated below for the reader's convenience. Generally, the mathematical transformation can be applied for the extended Kalman filter to obtain an Extended Information Filter (**EIF**). At each time step of measurement update, the filtering will evaluate the information vector contribution \mathbf{i}_k and its associated information matrix \mathbf{I}_k as in (3.31) and (3.32) [109], where $\nabla \mathbf{h}$ is the Jacobian of the measurement model. Then, the information vector $\hat{\mathbf{y}}_{k|k-1}$ and information matrix $\mathbf{Y}_{k|k-1}$ are updated as in (3.33).

$$\mathbf{i}_k = \nabla \mathbf{h}_x^T \mathbf{R}_k^{-1} (\tilde{\mathbf{z}}_k + \nabla \mathbf{h}_x^T \hat{\mathbf{x}}_{k|k-1}) \quad (3.31)$$

$$\mathbf{I}_k = \nabla \mathbf{h}_x^T \mathbf{R}_k^{-1} \nabla \mathbf{h}_x \quad (3.32)$$

$$\hat{\mathbf{y}}_{k|k} = \hat{\mathbf{y}}_{k|k-1} + \mathbf{i}_k \quad (3.33)$$

$$\mathbf{Y}_{k|k} = \mathbf{Y}_{k|k-1} + \mathbf{I}_k$$

Despite the mentioned improvement using the information matrix, EIF still shows drawbacks inherent in EKF: the linearization stability and the nontrivial nature of the Jacobian matrix employment [109]. The cubature Kalman filter approach is considered at this point

to overcome these issues. However, the CKF structure does not need to obtain the Jacobian matrix of the measurement model; it cannot be directly employed in the information filter's framework. Alternatively, the approximation step of the linear error propagation of the innovation covariance matrix \mathbf{P}_{zz} and the error cross covariance matrix \mathbf{P}_{xz} are utilized as in (3.34) [76, 109].

$$\begin{aligned}\mathbf{P}_{zz} &\simeq \nabla \mathbf{h}_x \mathbf{P}_{k|k-1} \nabla \mathbf{h}_x^T \\ \mathbf{P}_{xz} &\simeq \mathbf{P}_{k|k-1} \nabla \mathbf{h}_x^T\end{aligned}\tag{3.34}$$

$\mathbf{P}_{k|k-1}^{-1}$ and $\mathbf{P}_{k|k-1}$ are multiplied on the right hand side of (3.31) and (3.32) to have (3.35)), (3.37). Then, (3.34) is used to obtain (3.36) and (3.38). At each time step, the information vector contribution \mathbf{i}_k and its associated matrix \mathbf{I}_k are evaluated by (3.36) and (3.38) before correcting $\hat{\mathbf{y}}_{k|k-1}$ and $\mathbf{Y}_{k|k-1}$ by (3.33).

$$\begin{aligned}\mathbf{i}_k &= \mathbf{P}_{k|k-1}^{-1} \mathbf{P}_{k|k-1} \nabla \mathbf{h}_x^T \mathbf{R}_k^{-1} \\ &\quad (\tilde{\mathbf{z}}_k + \mathbf{P}_{k|k-1} \nabla \mathbf{h}_x^T \mathbf{Y}_{k|k-1}^T \hat{\mathbf{x}}_{k|k-1})\end{aligned}\tag{3.35}$$

$$\Rightarrow \mathbf{i}_k = \mathbf{Y}_{k|k-1} \mathbf{P}_{xz} \mathbf{R}_k^{-1} (\tilde{\mathbf{z}}_k + \mathbf{P}_{xz}^T \mathbf{Y}_{k|k-1}^T \hat{\mathbf{x}}_{k|k-1})\tag{3.36}$$

$$\mathbf{I}_k = \mathbf{P}_{k|k-1}^{-1} \mathbf{P}_{k|k-1} \nabla \mathbf{h}_x^T \mathbf{R}_k^{-1} \mathbf{P}_{k|k-1} \nabla \mathbf{h}_x^T \mathbf{P}_{k|k-1}^{-T}\tag{3.37}$$

$$\Rightarrow \mathbf{I}_k = \mathbf{Y}_{k|k-1} \mathbf{P}_{xz} \mathbf{R}_k^{-1} \mathbf{P}_{xz}^T \mathbf{Y}_{k|k-1}^T\tag{3.38}$$

For a system with d sensors, the CIF will evaluate the contribution of each sensor using (3.39) and (3.40). The synthesis of all contributions is operated with the summation form. This operation is more computationally efficient than the CKF and EKF structures, where the integration is in multiplication form. Figure 3.22 describes the system architecture of

CIF in the general case of multiple measurement updates.

$$\begin{aligned}\hat{\mathbf{y}}_{k|k} &= \hat{\mathbf{y}}_{k|k-1} + \sum_{j=1}^d \mathbf{i}_{j,k} \\ \mathbf{Y}_{k|k} &= \mathbf{Y}_{k|k-1} + \sum_{j=1}^d \mathbf{I}_{j,k}\end{aligned}\tag{3.39}$$

where

$$\begin{aligned}\mathbf{i}_{j,k} &= \mathbf{Y}_{k|k-1} \mathbf{P}_{j,xz} \mathbf{R}_k^{-1} (\tilde{\mathbf{z}}_k + \mathbf{P}_{j,xz}^T \mathbf{Y}_{k|k-1}^T \hat{\mathbf{x}}_{k|k-1}) \\ \mathbf{I}_{j,k} &= \mathbf{Y}_{k|k-1} \mathbf{P}_{j,xz} \mathbf{R}_k^{-1} \mathbf{P}_{j,xz}^T \mathbf{Y}_{k|k-1}^T\end{aligned}\tag{3.40}$$

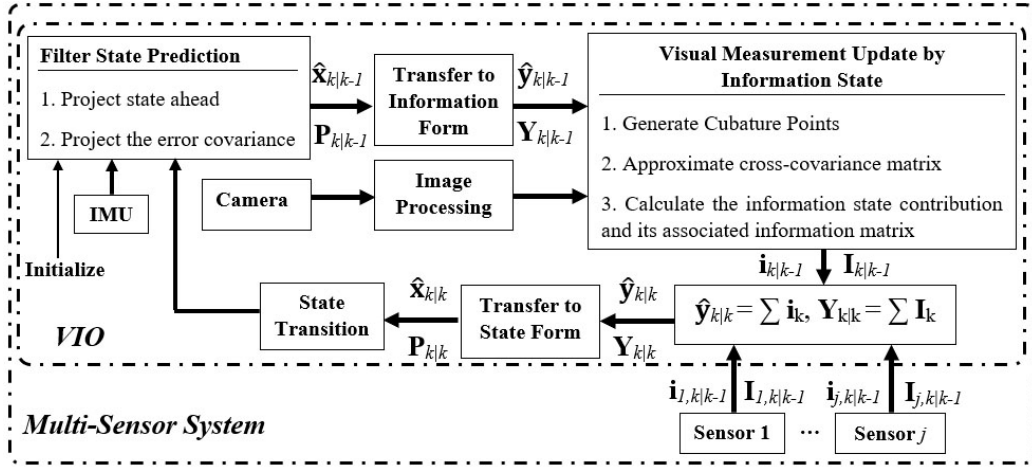


Figure 3.22: CIF system architecture with multiple measurement updates

It can be observed that the cubature information filter does not include the filter gain computation and the inverse of the innovation covariance matrix \mathbf{P}_{zz} , but requires the inverse of the covariance matrix \mathbf{P} to compute the information matrix. Our system has a 27×1 dimensional error filter state and f feature points. Each feature point has a 5×1 dimensional measurement model. The maximum dimension of matrix in the information filter to be inverted is the error filter state dimension ($\mathbf{P}_{27 \times 27}$). In the cubature Kalman filter, the \mathbf{P}_{zz} matrix will have dynamic dimensions of $5f \times 5f$. Generally, the observation

dimension is larger than the state dimension and it is expected to have more features to effectively correct the predicted filter state, $f \geq 20$. As a result, the inverse of \mathbf{P} involves less computational cost than the inverse of \mathbf{P}_{zz} and the system achieves computational efficiency.

3.6.2 Experiments for Cubature Information Filter

The CIF is validated with KITTI datasets. In the experiment, CIF is compared with CKF to evaluate the benefits of transforming the system to the information domain. The results are presented in Figs.3.23, 3.26, and 3.29 with different colours assigned for particular algorithms: cubature Kalman filter (**red**), cubature information filter (**blue**) and ground truth data (**black**). The Unscented Information Filter (**UIF**) (**green**) is also implemented in the experiment. UIF is the mathematical transformation of UKF into the information domain. Figs. 3.23, 3.26, and 3.29 present the tracking experimental results of the KITTI datasets drawn on a Google map while Figs.3.24, 3.27 and 3.30 provide RMSE evaluation of positional estimation. The proposed algorithms were able to track the vehicle trajectories with satisfactory accuracy against the ground truth data in different scenarios, especially with dataset 2011_09_30_0020 when the vehicle was making a U-turn, and dataset 2011_09_30_0033 when the vehicle was traveling with high velocity ($\sim 45\text{km/h}$) and for a long distance (about 1.8km). Figs.3.25, 3.28 and 3.31 evaluate the satisfactory rotation error of these filters. Also, the RMSE positional estimate error of CIF and CKF are almost similar in Figs. 3.27 and 3.30. The CKF achieves slightly better estimation accuracy than CIF in Fig. 3.24. When transforming into the information domain, the approximation of the cross-covariance matrix affects the estimation accuracy at an acceptable degree.

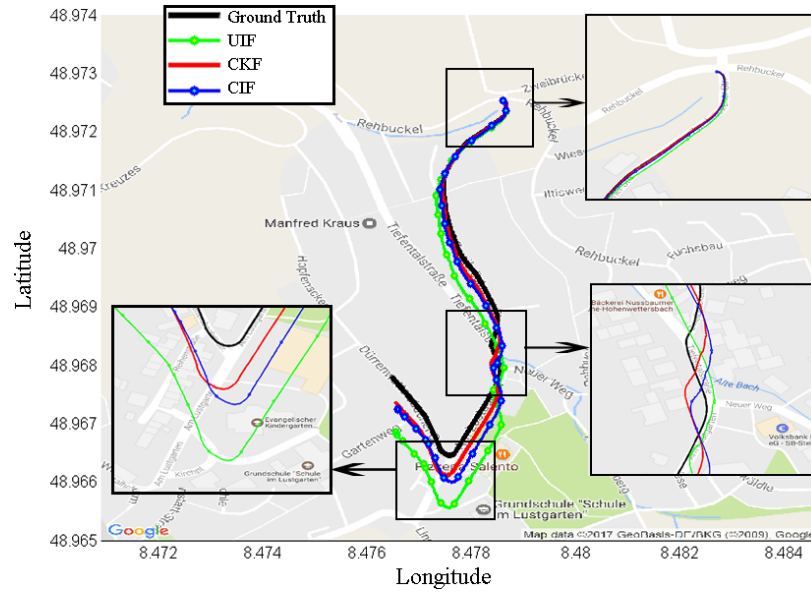


Figure 3.23: Experimental results drawn on Google map using dataset 2011_09_30_0034

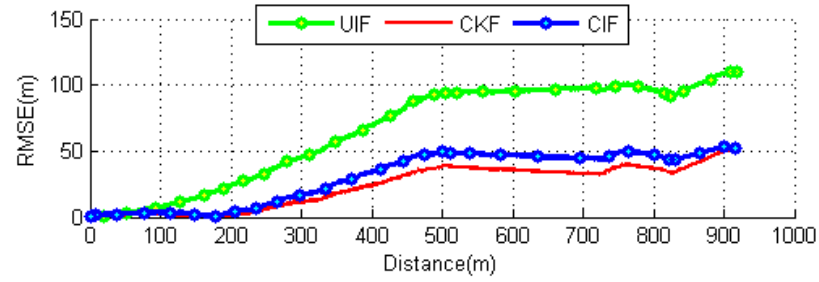


Figure 3.24: RMSE of positional estimation with KITTI dataset 2011_09_30_0034

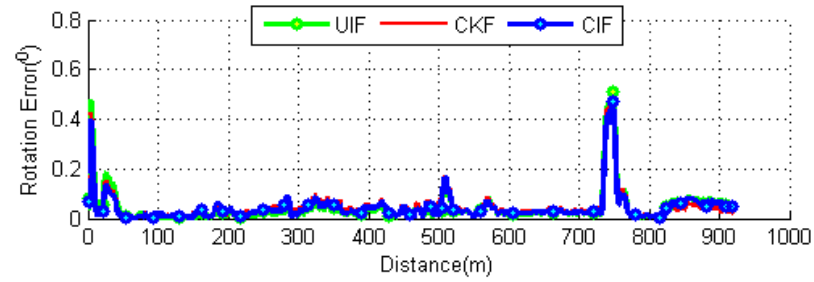


Figure 3.25: Rotation error evaluation of KITTI dataset 2011_09_30_0034

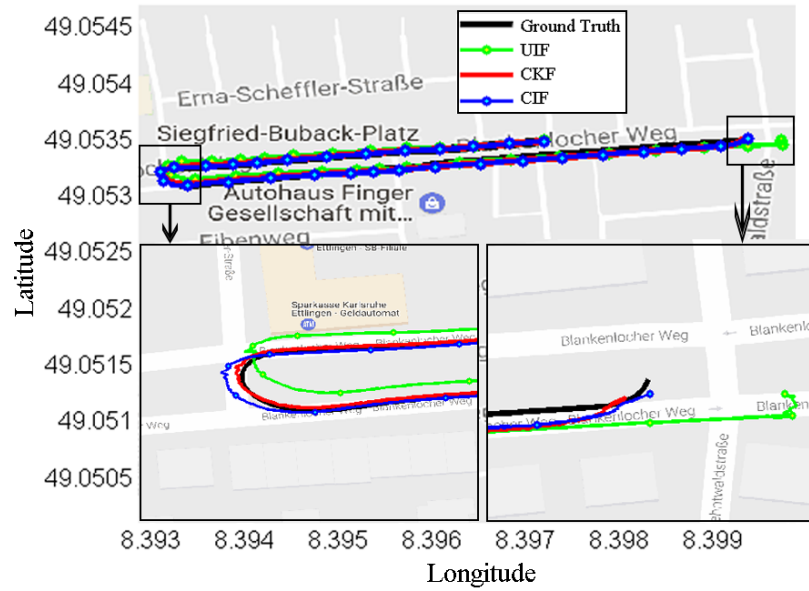


Figure 3.26: Experimental results drawn on Google map using dataset 2011_09_30_0020

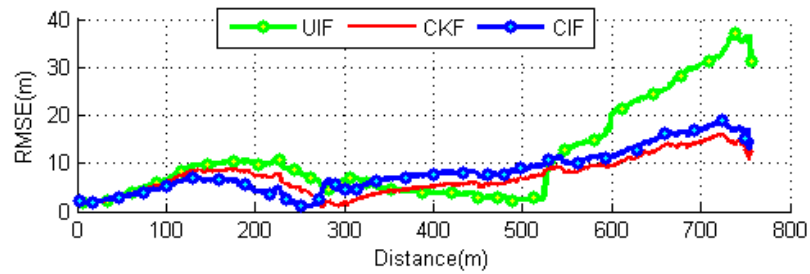


Figure 3.27: RMSE of positional estimation with KITTI dataset 2011_09_30_0020

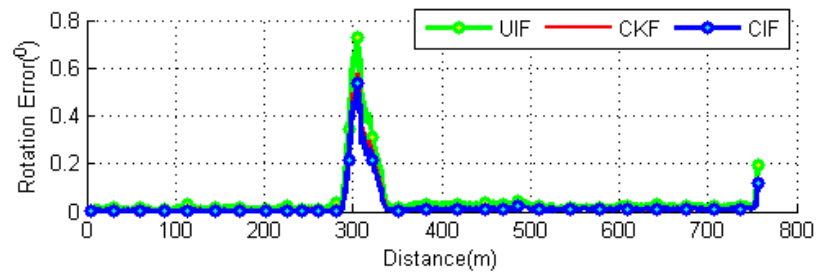


Figure 3.28: Rotation error evaluation of KITTI dataset 2011_09_30_0020

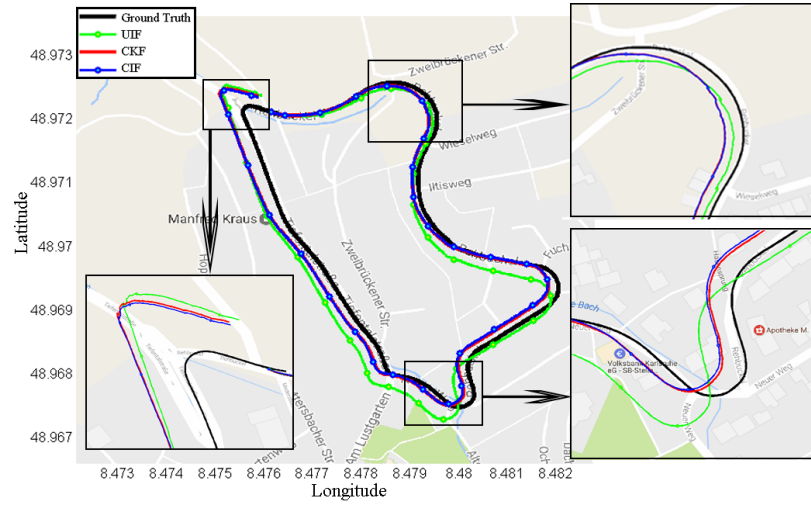


Figure 3.29: Experimental results drawn on Google map using dataset 2011_09_30_0033

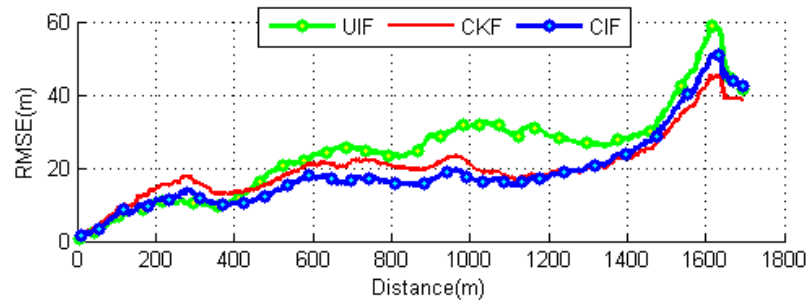


Figure 3.30: RMSE of positional estimation with KITTI dataset 2011_09_30_0033

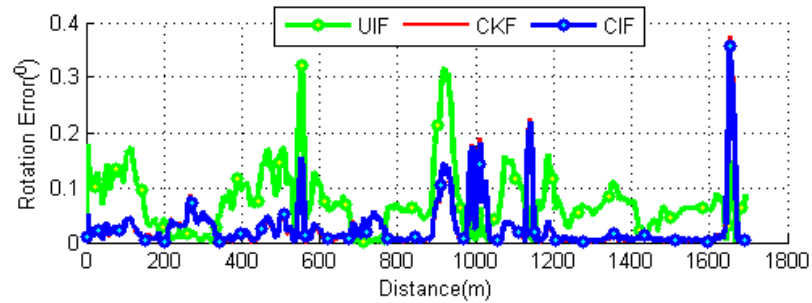


Figure 3.31: Rotation error evaluation of KITTI dataset 2011_09_30_0033

In the experiment, the CIF achieves better estimation accuracy than the UIF. In reality, the UIF is developed from UKF, which shows some limitations in handling negative sampling points and the square-root operation of the covariance matrix [18]. CKF is designed as a better solution to handle these limitations, which improves the system stability and the estimation accuracy. When the CIF is developed from CKF, CIF also inherits these improvements. In Fig. 3.23, these filters only produce accurate estimates within the first distance of 400m. After that, the error accumulation in CIF is similar to that of CKF. The employment of additional sensors in the scheme of the multi-sensor system will help to minimize the error accumulation. The structure of the information filter accelerates the computation of multiple measurement updates.

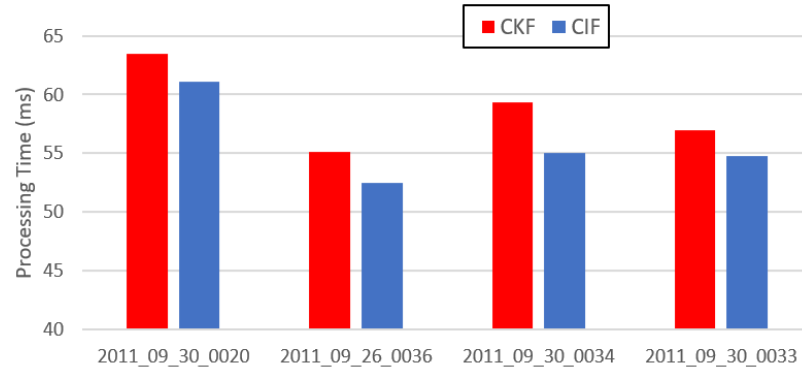


Figure 3.32: Processing time evaluation of CKF and CIF

Fig.3.32 provides the processing time measurement of executing the visual measurement update step for 50 tracked features. The CIF takes less processing time ($\sim 4\%$) to update than the CKF. The transformation to the information domain for decreasing the computation cost is verified. Also, the CIF implementation allows the independent evaluation of each feature's contribution during the measurement update. This benefit helps

to relax the assumption of the visual measurement noise, where all landmark features are treated equally. The video demonstration of the CIF can be located at the following link: youtu.be/8udBBHZ9NI8.

3.7 Conclusions

This chapter described the development of a computationally efficient VIO algorithm which combined the TTG-based visual measurement model and CKF. Multiple experiments used publicly available datasets to validate the proposed algorithm. Compared to a traditional VIO filtering algorithm such as EKF, which extracts 3D feature points during the measurement update, the proposed algorithm with TTG used fewer camera images and computational resources. The processing time of the measurement prediction step was reduced by $\sim 95\%$ relative to the traditional computation. The CKF implementation helps to handle the nonlinearity efficiently, which produces better estimation accuracy than EKF in VIO application.

This chapter also highlighted the computationally inefficiency of the Kalman structure when the VIO system processes considerable amounts of visual data. The mathematical transformation of CKF to the information domain helps to further reduce the processing times by $\sim 4\%$, while maintaining a similar computational error. Overall, the proposed algorithm can be implemented as a fast VIO solution for micro and nano robots. Additionally, the CIF structure helps to evaluate the contribution of each feature independently in the measurement update step. This benefit motivates the employment of parallel processing in VIO application in the following chapter.

Chapter 4

Accurate Visual Inertial Odometry for Long-Term Trajectory Operations Without Using a Map

The estimation error accumulation in the conventional VIO generally does not produce accurate long-term operation. To address this issue, two solutions, as described in this chapter¹, were developed. The first solution implements an iterated cubature Kalman filter, which performs multiple corrections on a single measurement to optimize the current

¹This chapter is based on the following papers of the author:

- T. Nguyen, G. K. I. Mann, A. Vardy, and R. G. Gosine, "Likelihood-based Iterated Cubature Multi-State-Constraint Kalman Filter for Visual Inertial Navigation System", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017
- T. Nguyen, G. K. I. Mann, A. Vardy, and R. G. Gosine, "CKF-Based Visual Inertial Odometry for Long-Term Trajectory Operations", ASME Journal of Dynamic Systems, Measurement and Control (Under Review), 2019

filter state and covariance during the measurement update. The optimization process is terminated using the maximum likelihood estimate based criteria. The second solution implements CKF with additional employment of pseudo-ranging measurement between the vehicle and multiple anchors. The integration follows the sequential-sensor-update approach, which in turn enables an independent operation between the ranging sensors and VIO system.

4.1 Introduction

Visual inertial odometry employs the sensor fusion between IMU measurements and camera's image information to enhance the accurate estimation of the vehicle trajectory [68, 125]. Analyzing the VIO system architecture, the front-end and back-end computations are designed to exploit the benefits of both sensors, capture the system uncertainty, produce reliable ego-motion estimation and achieve robust performance. The vision front-end computation attempts to track 3D feature points, which is used to correct the prediction from the IMU data. The effectiveness of the correction depends on the sensor-fusion strategy. Multiple strategies have been introduced with varying requirements of hardware computing resources. Such solutions include MSCKF (version 1.0 [95] and version 2.0 [79]), VINS-Mono [119] and OKVIS [78].

VIO systems suffer from the inevitable accumulation of error. This limitation makes the filter gradually diverge and even fails to track the vehicle trajectory over long-term operation. VIO only produces reliable estimation of the vehicle trajectory in short-term operation and short-distance travel. This issue obviously demands the development of VIO techniques to minimize estimation error over long-term operation. Recent literature has

introduced multiple approaches to reduce the accumulative error. These approaches include global pose graph optimization [82] and loop closure [78,82]. These techniques require the entire trajectory and map to re-localize the vehicle estimates and improve the estimation accuracy. Such solutions demand extremely high computational load, memory utilization and processing time for execution. These costs are challenging in real-time computation and may not be affordable for some micro-robotic systems [44, 68, 158], having limited hardware computing capability. For example, Delmerico et al. [44] reported the failure to execute these advanced techniques of VINS-Mono [119] on ODROID, an embedded PC with a hybrid processing unit [8]. Moreover, these solutions are effective only when the vehicle makes a closed-loop trajectory to re-observe some previous landmark features. Not all vehicle trajectories include loops to activate the loop closure. Alternatively, other researches [23, 71] have performed a local optimization within the sliding window during the measurement update step. Their implementations did not require the installation of any specialized hardware computing platform or advanced optimization software library. The effectiveness of this technique depends on the tuning parameters and the criteria to terminate the iteration.

Having developed the tightly-coupled VIO filtering algorithm in the previous chapter, the research described in this chapter is attempting to upgrade the VIO estimation, which allows the filter to operate for a longer duration. This chapter proposes two filtering solutions, which focus on the inference of the vehicle's latest state rather than the entire trajectory. For the first solution, an iterated CKF and TTG [60] based measurement model for the VIO estimation is applied. An iteration procedure is enhanced to perform multiple corrections on a single measurement. The process locally optimizes the estimate of the current filter state and covariance during the visual measurement update. The system em-

employs maximum likelihood estimate based criteria to terminate this optimization. For the second solution, pseudo-ranging measurements between the vehicle and multiple anchors are implemented. The design of the integration follows the sequential-sensor-update approach, which enables operating independence between each sensor. This property allows the system to operate even without any ranging measurement.

To summarize, this chapter describes research that makes the following contributions. Firstly, two novel solutions associated with the VIO filtering technique to improve the estimation accuracy over long-term operation without using any map or the entire trajectory are developed. Secondly, the benefit of the TTG model to enhance the optimization during the filter update step is exploited. The implementation does not increase the system complexity significantly or require the installation of any advanced optimization library. This strategy is suitable for self-localization projects without using any additional sensor. Thirdly, the combination of VIO and pseudo-ranging measurement to bound the estimation error over long-term operation is implemented. This solution can be applied for large-scale navigation projects. The remainder of the chapter is organized as follows. The next section presents the issue of error accumulation. Section 4.3 describes the first solution employing iterated CKF, while section 4.4 describes the second solution using ranging sensors. Experimental validation for each solution is also reported accordingly. Finally, section 4.5 presents some discussion and conclusions.

4.2 CKF-based VIO's Error-Accumulation Issue

After developing the CKF-based VIO in the previous chapter, this section illustrates the error accumulation issue. For our purpose, the experimental results with the KITTI dataset

[56] in the previous chapter are reused. The estimation is inevitably subject to error accumulation (drift) like any other odometry estimate. Fig.4.1 presents the VIO estimation when the vehicle travels a long distance. As in Fig.4.1, Root-Mean-Square Error is accumulated gradually. The estimation drifts from its real trajectory, which can be observed clearly in Fig.4.1. In Fig.4.1 with dataset 2011_09_30_0034, the VIO estimation drifts about 50m after traveling 900m. Similarly, the VIO estimation drifts about 45m after traveling 1600m with dataset 2011_09_30_0033 in Fig.4.2.

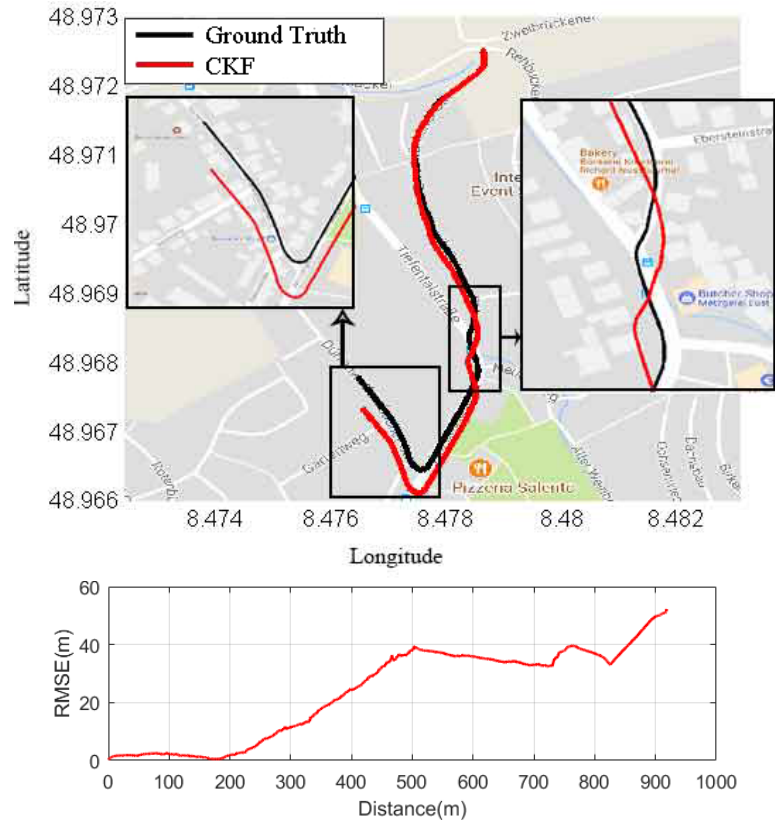


Figure 4.1: Position estimation in dataset 2011_09_30_0034

The CKF-based system has employed IMU data for the filter state prediction and camera images for the filter correction. The drift is mainly derived from the poor performance

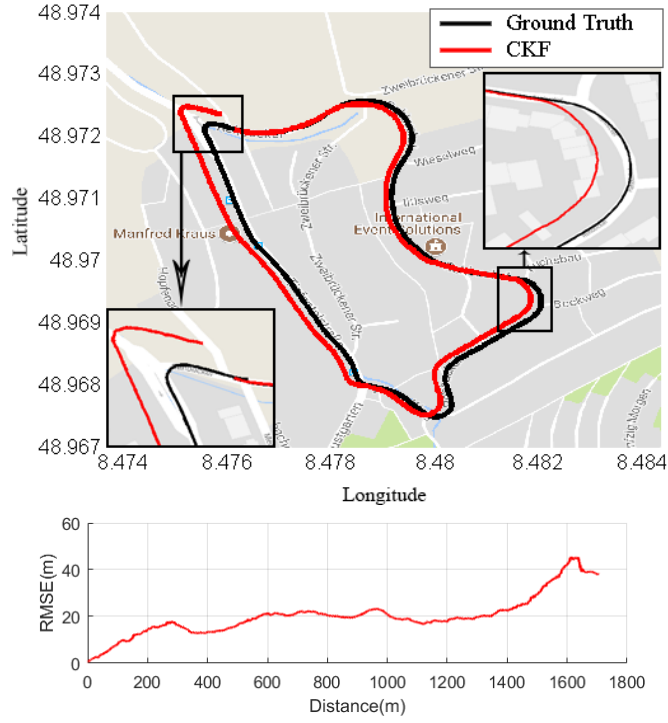


Figure 4.2: Position estimation in 2011_09_30_0033

of the camera observation. The visual measurement update step is unable to correct the residuals completely and suppress the drift effectively. The error accumulation can come from various sources:

- (i) The camera resolution and calibration is limited to provide reliable measurements in some particular cases such as traveling too fast.
- (ii) The front-looking monocular camera has limited depth perception.
- (iii) The sensor-fusing algorithm is unable to fully capture the uncertainties and eliminate the environment noise.

The first two issues (i - ii) are associated with the hardware configuration, while the last issue (iii) is derived from the sensor-fusing technique. Additionally, the drift increases

unboundedly for an assumed unlimited time. The drift in orientation, which is limited to the range $[-\pi, \pi]$, also contributes to the drift in position. For short-term consideration (about 1 second), the drift can be modeled as a linear motion to simplify the problem formation [154]. In long-term consideration, the drift does not grow linearly in the traveled distance. It can be treated as a random process and increases or decreases depending on the errors in motion vectors [84]. Assuming that the modification of the hardware setup is not an optimal solution, a sensor-fusing algorithm is developed, as described in the following section, with two proposed solutions.

4.3 Solution 1: Iterated Cubature Kalman Filter

4.3.1 Maximum Likelihood Estimate Based Optimization

The previous chapter has introduced the CKF design for VIO application. Solution 1 upgrades the visual measurement update to conduct the local optimization. The CKF and UKF share a common property (i.e. using a weighted set of symmetric points to handle the nonlinearity). This chapter utilizes the design of UKF based local optimization presented in the work [143] in order to develop the iterated CKF.

Similarly, $\mathbf{S}_{k|k-1}$ is the square-root factor of the covariance $\mathbf{P}_{k|k-1}$, which satisfies the equality $\mathbf{P}_{k|k-1} = \mathbf{S}_{k|k-1} \mathbf{S}_{k|k-1}^T$. Cubature points ($s = 1, 2, \dots, 2n$) are generated with their particular parameters (4.1) and (4.2).

$$\xi_s = \begin{cases} \sqrt{n}e_s & s = 1, 2, \dots, n \\ -\sqrt{n}e_{s-n} & s = n+1, n+2, \dots, 2n \end{cases} \quad (4.1)$$

$$\mathbf{X}_{s,k|k-1} = \mathbf{S}_{k|k-1} \xi_s + \hat{\mathbf{x}}_{k|k-1} \quad (4.2)$$

where $e_s \in \mathbb{R}^{n \times 1}$ represents the s^{th} elementary column vector. These cubature points are evaluated through the nonlinear measurement model to obtain the predicted measurements (4.3). Then, the innovation covariance matrix \mathbf{P}_{zz} and the cross covariance matrix \mathbf{P}_{xz} can be computed as in (4.4).

$$\begin{aligned}\mathbf{Z}_{s,k|k-1} &= \mathbf{h}(\mathbf{X}_{s,k|k-1}, \{\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3\}) \\ \hat{\mathbf{z}}_{s,k|k-1} &= \sum_{s=1}^{2n} \frac{1}{2n} \mathbf{Z}_{s,k|k-1}\end{aligned}\tag{4.3}$$

$$\begin{aligned}\mathbf{P}_{xz} &= \sum_{s=1}^{2n} \frac{1}{2n} (\mathbf{X}_{s,k|k-1} - \hat{\mathbf{x}}_{s,k|k-1}) (\mathbf{Z}_{s,k|k-1} - \hat{\mathbf{z}}_{s,k|k-1})^T \\ \mathbf{P}_{zz} &= \sum_{s=1}^{2n} \frac{1}{2n} (\mathbf{Z}_{s,k|k-1} - \hat{\mathbf{z}}_{s,k|k-1}) (\mathbf{Z}_{s,k|k-1} - \hat{\mathbf{z}}_{s,k|k-1})^T\end{aligned}\tag{4.4}$$

The iteration procedure is conducted with the TTG-based visual measurement to optimize the estimated state and covariance. Generally, the estimated state $\hat{\mathbf{x}}_{k|k}$ is closer to the filter's true state than the predicted state $\hat{\mathbf{x}}_{k|k-1}$. In the iteration, the estimated state $\hat{\mathbf{x}}_{k|k}^{(j)}$ at j^{th} iteration produces a better approximation to the filter true state than the estimated state $\hat{\mathbf{x}}_{k|k}^{(j-1)}$ at $(j-1)^{th}$ iteration. The state correction is performed as in (4.5):

$$\hat{\mathbf{x}}_{k|k}^{(j)} = \hat{\mathbf{x}}_{k|k}^{(j-1)} + \mathbf{K}_k^{(j)} (\mathbf{z}_k - \hat{\mathbf{z}}_k)\tag{4.5}$$

$$\mathbf{K}_k^{(j)} = \frac{\mathbf{P}_{xz}^{(j)}}{\mathbf{P}_{zz}^{(j)} + \mathbf{R}_c}\tag{4.6}$$

$$\mathbf{P}_{k|k}^{(j)} = \mathbf{P}_{k|k}^{(j-1)} - \mathbf{K}_k^{(j)} \mathbf{P}_{zz}^{(j)} (\mathbf{K}_k^{(j)})^T\tag{4.7}$$

When the iteration is executed, the system can compute the positive-definite covariance matrices $\mathbf{P}_{k|k}^{(j)}$, $\mathbf{P}_{k|k}^{(j-1)}$ and $\mathbf{P}_{zz}^{(j)}$. Assuming $\lim_{j \rightarrow \infty} \mathbf{K}_k^{(j)} \neq 0$, equation (4.7) reveals $\mathbf{P}_{k|k}^{(j)} < \mathbf{P}_{k|k}^{(j-1)}$ for any $j = 1, 2, \dots, \infty$. However, when each element of the matrix $\mathbf{P}_{k|k}^{(j)}$ is bounded, it also results in $\lim_{j \rightarrow \infty} \mathbf{P}_{k|k}^{(j)} = \lim_{j \rightarrow \infty} \mathbf{P}_{k|k}^{(j-1)}$ [143]. Additionally, it can also be inferred from (4.7) that

$\lim_{j \rightarrow \infty} \mathbf{K}_k^{(j)} = 0$, which violates our initial assumption ($\lim_{j \rightarrow \infty} \mathbf{K}_k^{(j)} \neq 0$). Actually, that assumption cannot hold with iteration. Therefore, $\lim_{j \rightarrow \infty} \mathbf{K}_k^{(j)} = 0$. Consequently, when $\mathbf{K}_k^{(j)} \rightarrow 0$ with $j > N$, $\mathbf{x}_{k|k}^{(j)} \rightarrow \mathbf{x}_{k|k}^{(j-1)}$ and $\mathbf{P}_{k|k}^{(j)} \rightarrow \mathbf{P}_{k|k}^{(j-1)}$. In other words, the global convergence is guaranteed during the iteration procedure.

$$\Delta \hat{\mathbf{x}}_{k|k}^{(j)} = \hat{\mathbf{x}}_{k|k}^{(j)} - \hat{\mathbf{x}}_{k|k}^{(j-1)} \quad (4.8)$$

When the state is converging, the state variation $\Delta \hat{\mathbf{x}}_{k|k}^{(j)}$ can be evaluated as in (4.8). Obviously, a predetermined threshold ρ can be set for $\Delta \hat{\mathbf{x}}_{k|k}^{(j)} < \rho$ to terminate the iteration. However, tuning that threshold ρ is challenging. Although the iteration with the threshold ρ can achieve global convergence, it does not guarantee that the optimal value would be in the likelihood surface.

Assuming $\mathbf{x}_{k|k} \sim N(\hat{\mathbf{x}}_k^{(j)}, \mathbf{P}_{k|k}^{(j)})$ and $\mathbf{z}_k \sim N(\hat{\mathbf{z}}_k, \mathbf{R}_k)$, the cost function is determined within the structure of the CKF in order to terminate the iteration:

$$J(\mathbf{x}_k^{(j)}) = (\Delta \hat{\mathbf{x}}_{k|k}^{(j)})^T (\mathbf{P}_{k|k}^{(j)})^{-1} \Delta \hat{\mathbf{x}}_{k|k}^{(j)} + (\Delta \mathbf{z}_k^{(j)})^T \mathbf{R}_k^{-1} \Delta \mathbf{z}_k^{(j)} \quad (4.9)$$

where

$$\Delta \mathbf{z}_k^{(j)} = \mathbf{z}_k - \mathbf{z}_k^{(j)} \quad (4.10)$$

The minimum value of the cost function J can be found to determine the MLE of \mathbf{x}_k and \mathbf{z}_k through an optimization process, which is complicated and impractical [97]. However, in the process of the optimization, the inequality condition (4.11) is always satisfied. In other words, $J(\mathbf{x}_k^{(j)})$ is closer to the maximum likelihood surface than $J(\mathbf{x}_k^{(j-1)})$. Consequently, $\mathbf{x}_k^{(j)}$ is a more accurate approximation to MLE than $\mathbf{x}_k^{(j-1)}$.

$$J(\mathbf{x}_k^{(j)}) < J(\mathbf{x}_k^{(j-1)}) \quad (4.11)$$

That inequality can be used to terminate the iteration. To apply our cubature measurement update with $\mathbf{P}_{k|k}^{(j)} = \mathbf{S}_{k|k}^{(j-1)} \mathbf{S}_{k|k}^{(j-1)}$, the inequality condition is rewritten as in (4.12) [143]:

$$(\Delta \hat{\mathbf{x}}_{k|k}^{(j)})^T (\mathbf{S}_{k|k}^{(j-1)} \mathbf{S}_{k|k}^{(j-1)})^{-1} \Delta \hat{\mathbf{x}}_{k|k}^{(j)} + (\Delta \mathbf{z}_k^{(j)})^T \mathbf{R}_k^{-1} \Delta \mathbf{z}_k^{(j)} < (\Delta \mathbf{z}_k^{(j-1)})^T \mathbf{R}_k^{-1} \Delta \mathbf{z}_k^{(j-1)} \quad (4.12)$$

4.3.2 The System Architecture of Iterated CKF

To summarize, the computation of iterated CKF is presented in the algorithm 1. Figure 4.3 describes the VIO system architecture developed with iterated CKF. The IMU data is employed for the filter state and covariance propagation. The camera observation is used for visual measurement update. The block *Image Processing* executes the visual front-end computation. The visual measurement step will be conducted with j^{th} iteration under the inequality condition (4.12) and maximum iteration $j < N$. If only one iteration is performed, the iterated CKF becomes the CKF. The *State Transition* block guarantees the appropriate filter state and image transition after finishing the measurement update.

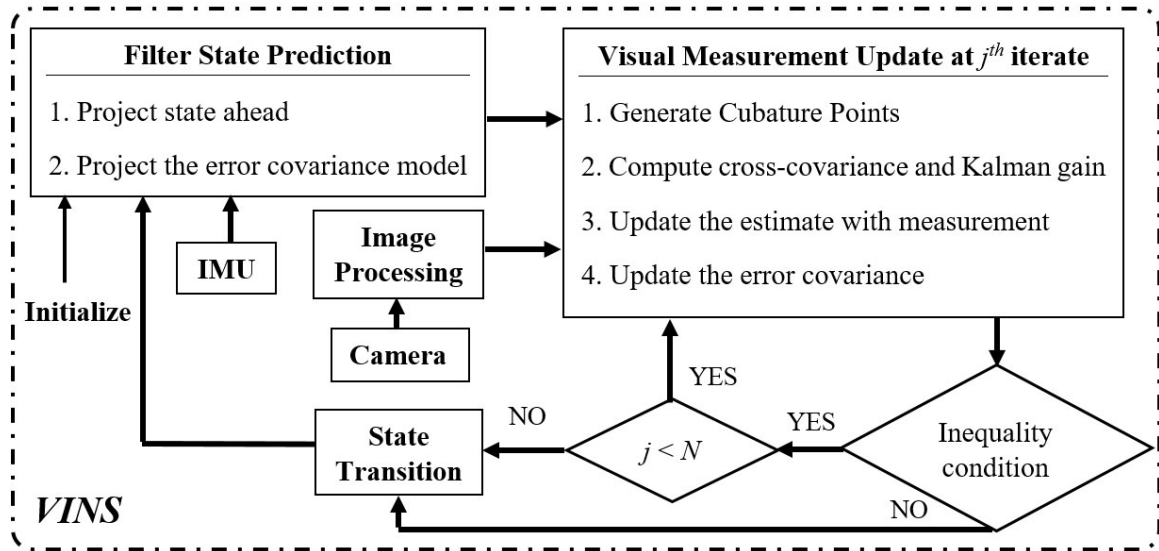


Figure 4.3: The system architecture developed with iterated CKF

Algorithm 1: Iterated CKF Algorithm

```
1 Filter initialization with initial state and initial covariance matrix ;
2 for  $k \in (1, \dots, \infty)$  do
3   Predict the nominal state  $\hat{\mathbf{x}}_k$  by IMU data using 4th order Runge-Kutta
   numerical integration;
4   Calculate  $\mathbf{F}_d$  and  $\mathbf{Q}_d$  ;
5   Compute the propagated state covariance  $\mathbf{P}_{k|k-1}$  ;
6   if New Image  $I_k$  then
7      $I_k \rightarrow I_3$ . Perform feature detection on  $I_3$  and match these features with
     features  $\{m_1\}$  and  $\{m_2\}$  to have  $\{m_1\} \leftrightarrow \{m_2\} \leftrightarrow \{m_3\}$ ;
8     Factorize:  $\mathbf{P}_{k|k-1} = \mathbf{S}_{k|k-1} \mathbf{S}_{k|k-1}^T$  ;
9     while ((Condition (4.12) is satisfied) and ( $j < N$ )) do
10      Calculate the variation rate  $\Delta \mathbf{x}_{k|k}^{(j)}$ ,  $\Delta \mathbf{z}_{k|k}^{(j)}$  by (4.8), (4.10), respectively;
11      Generate cubature points by (4.2) ;
12      Calculate the predicted measurement and innovation covariance matrix:
      (4.3)-(4.4);
13      Update state and associated covariance matrix (4.6)-(4.7) ;
14    end
15    Perform state transition and revise the covariance matrix;
16  end
17 end
```

4.3.3 Experimental Validation of Solution 1

The KITTI dataset is also used for experimental validation because it allows verifying the system performance for long-term operations with long-distance traveling. It also helps to observe the impact of the accumulative errors as well as the effect of implementing each solution. The estimates of these filters are presented in a GoogleTMmap with multiple zoom-in images for comparison. The evaluation criteria are RMSE and the rotation error. The ground truth data are obtained from GPS/IMU Inertial Navigation System (INS) data.

Figures 4.4, 4.5 and 4.6 present the experiment with dataset 2011_09_26_0087. The vehicle travels about 290m. The UKF has failed to track the vehicle trajectory after 50m. The iterated CKF has less RMSE of the position estimate than the CKF. In the rotation estimate (Fig.4.6), the estimates of Iterated CKF and the CKF are comparative. As expected, the INS system with only the integration of IMU data is not able to track the vehicle trajectory. The fusion with the camera has improved the quality of the estimation.

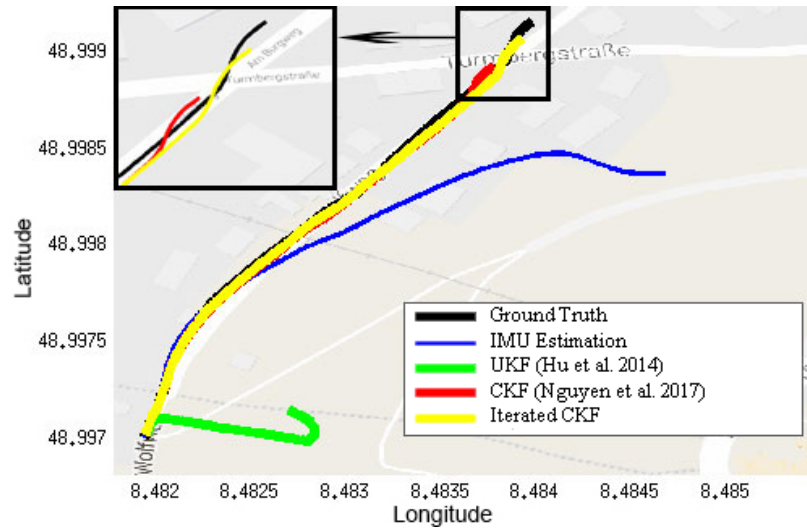


Figure 4.4: Experimental results presented in Google map for 2011_09_26_0087

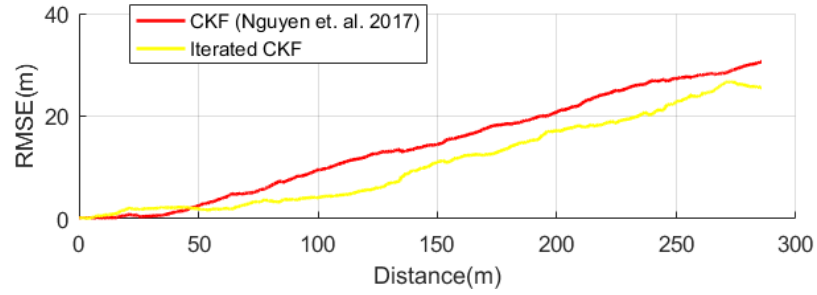


Figure 4.5: RMSE evaluation in the experiment with dataset 2011_09_26_0087

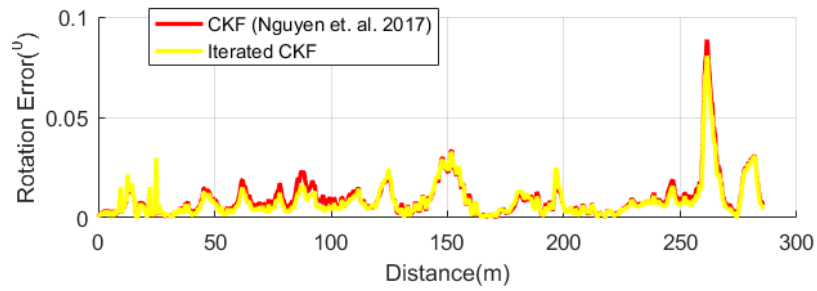


Figure 4.6: Rotation errors in the experiment with dataset 2011_09_26_0087

Figures 4.7, 4.8 and 4.9 present the experiment with dataset 2011_09_30_0034. The vehicle travels about 900m. In this case, the iterated CKF has superior performance than the other filters. The CKF and UKF produce accurate estimates within the first distance of 200m. The employment of the optimization process has effectively decreased the estimation error, somehow partly eliminating the accumulative errors. In Fig.4.7, the drift in the performance of iterated CKF after 800m. In Fig.4.9, the rotation estimates of these filters have similar accuracy. The iterated CKF continuously updates the bias of accelerometer and gyroscope at each time instant as in Fig.4.10.

Figures 4.11, 4.12 and 4.13 present the experiment with dataset 2011_09_30_0033. The vehicle travels about ~ 1700 m. The employment of iteration does not help to significantly improve the estimation accuracy. In other words, the optimization step with MLE can-

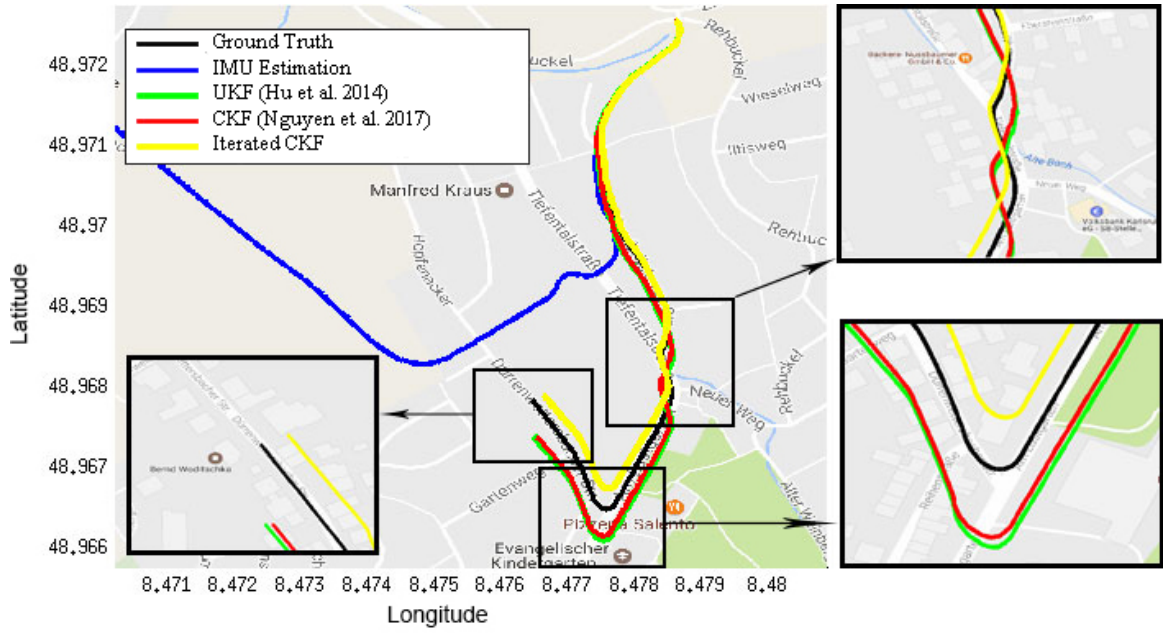


Figure 4.7: Experimental results presented in Google map for 2011_09_30_0034

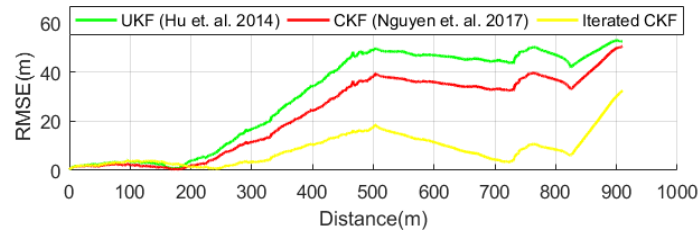


Figure 4.8: RMSE in the experiment with dataset 2011_09_30_0034

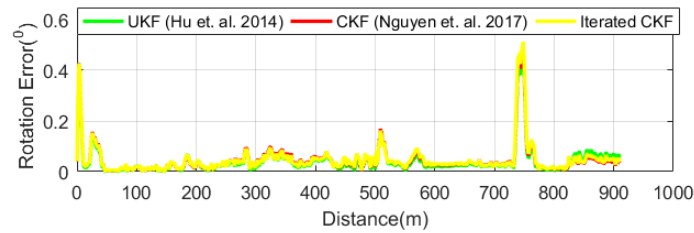


Figure 4.9: Rotation errors in the experiment with dataset 2011_09_30_0034

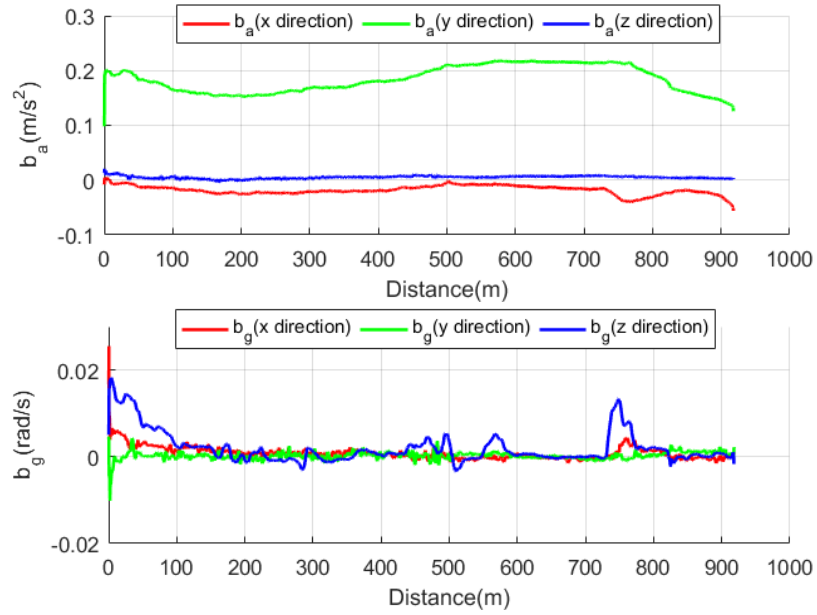


Figure 4.10: Iterated CKF estimation of accelerometer and gyroscope bias with dataset 2011_09_30_0034

not bring the estimate up to the likelihood surface. Considering the rotation accuracy in Fig.4.13, all three filters have almost similar accuracy. The video demonstration of this solution can be located at youtu.be/-8SWh-cy-Ck.

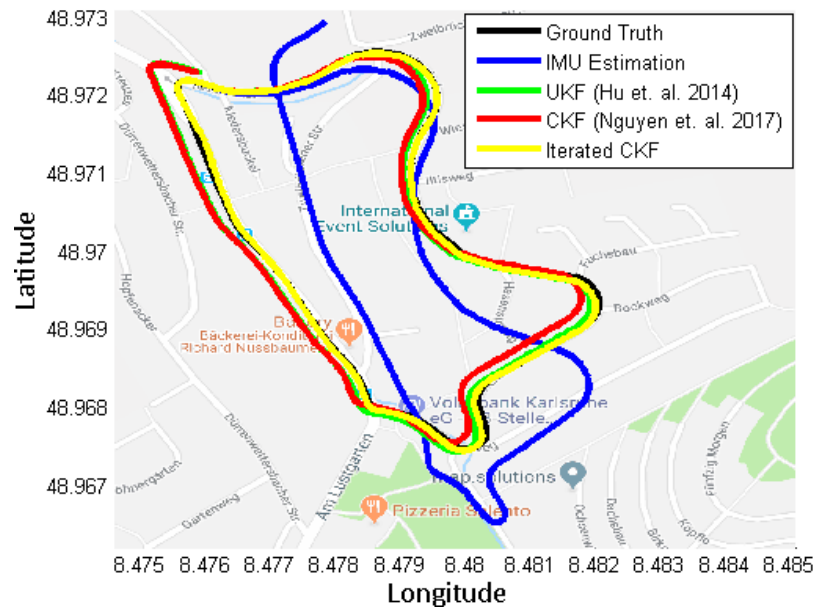


Figure 4.11: Experimental results presented in Google map for 2011_09_30_0033

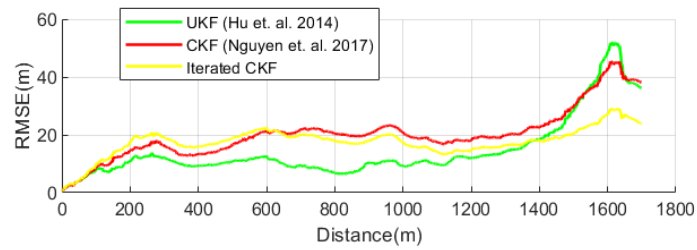


Figure 4.12: RMSE in the experiment with dataset 2011_09_30_0033

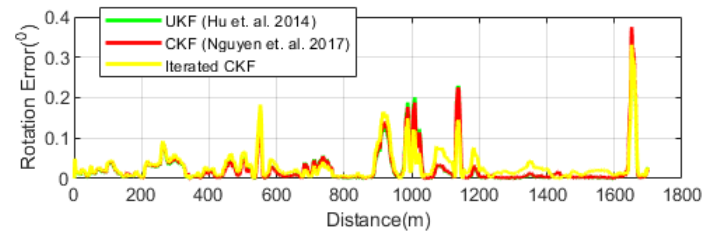


Figure 4.13: Rotation errors in the experiment with dataset 2011_09_30_0033

4.4 Solution 2: Pseudo-range Measurements

In solution 2, pseudo-range measurements are tightly integrated with VIO, which will bound the estimation error and correct the positional drifts. The pseudo-range measurement can be established by the wireless transmission between the anchor (or beacon) and tag units. The tag unit is mounted on the vehicle and communicates with multiple anchor units, which are installed rigidly at known locations in the environment. The vehicle can be passive [75] or active [98] in the communication process, depending on how many vehicles are employed. Each pseudo-range measurement can be modeled as in (4.13):

$$z_{r,d} = ||^G\mathbf{p}_{a,d} - ^G\mathbf{p}_I||_2 + \zeta_d\beta \quad (4.13)$$

where $^G\mathbf{p}_{a,d}$ is the position of the d^{th} anchor, $^G\mathbf{p}_I$ is the current position of the vehicle, β is a bias of range error model, ζ_d is the coefficient describing the influence of β on the pseudo-range measurement, $||.||_2$ is the Euclidean distance. Considering the real-time implementation, multiple hardware modules can be employed such as Decawave [3] or Time Domain's P-410 UWB module [15].

4.4.1 Sequential-Sensor-Update Approach

Multiple anchors are placed along the vehicle trajectory as in Fig.4.14. In reality, multiple pseudo-range measurements can enable the vehicle to self-localize by using either Time Of Arrival (**TOA**) or Time Difference of Arrival (**TDOA**) measurements. In general, this self-localization system requires at least four available ranging measurements to estimate accurately [75, 144]. However, if the vehicle is traveling a long distance, it is difficult to always receive enough ranging measurements to perform self-localization. Research

described in this chapter considers the case when the ranging measurements are only used to supplement the VIO.

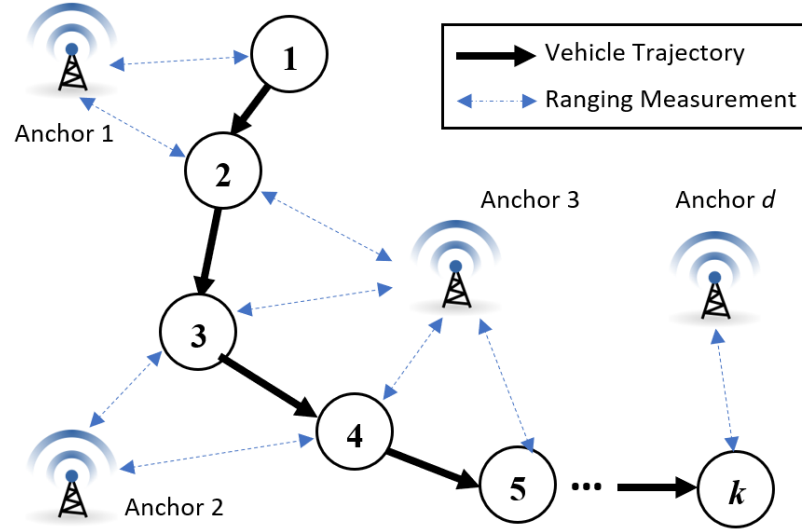


Figure 4.14: Illustration of the vehicle in wireless communication process for pseudo-range measurements

All measurements (i.e., visual and ranging measurements) can be synthesized into a single composite group sensor with only one measurement model and similarly apply CKF for the estimation. However, this group sensor approach assumes that all sensors have similar update rates and that measurements are always available. This assumption does not satisfy our system configuration when the camera and ranging sensors operate independently. The number of available ranging sensors may vary depending on the communication process. As a result, the sequential-sensor method is applied in this research. The sequential sensor update approach considers each sensor's observation as a separate and independent realization. Each sensor will operate following a specific observation model, which can be incorporated into the filter operation in a sequential manner [47]. A set of all observations

made by the camera up to time k will be denoted by $\mathbf{Z}_c^k \triangleq \{\mathbf{z}_c(1), \mathbf{z}_c(2), \dots, \mathbf{z}_c(k)\}$ and a set of all observations made by the pseudo-range sensor up to time k will be denoted by $\mathbf{Z}_r^k \triangleq \{\mathbf{z}_r(1), \mathbf{z}_r(2), \dots, \mathbf{z}_r(k)\}$. Hence, the set of all observations made by two sensors (camera and pseudo-range) up to time k is constructed by: $\mathbf{Z}_{c,r}^k \triangleq \{\mathbf{Z}_c^k \cup \mathbf{Z}_r^k\}$

The state prediction, computed by IMU data, is $\hat{\mathbf{x}}_{k|k-1}$ with covariance $\mathbf{P}_{k|k-1}$. The camera observation will update the filter state to $\mathbf{x}_{k|k}$ with covariance $\mathbf{P}_{k|k}$. The filter state after the camera measurement update can be denoted as $\mathbf{x}_{c,k|k}$ with covariance $\mathbf{P}_{c,k|k}$. The Kalman gain $\mathbf{K}_{r,k}$ is calculated with variance $\mathbf{R}_r = \text{diag}[v_r, v_r, v_r]$ as in (4.14). The pseudo-range innovation is computed as (4.15) before performing the correction step (4.16). $\hat{\mathbf{x}}_{k|k}$ with $\mathbf{P}_{k|k}$ are the estimate of VIO after the visual and pseudo-range measurement update.

$$\mathbf{K}_{r,k} = \frac{\mathbf{P}_{c,k|k} \mathbf{H}_{r,k}^T}{\mathbf{H}_{r,k} \mathbf{P}_{c,k|k} \mathbf{H}_{r,k}^T + \mathbf{R}_r} \quad (4.14)$$

$$\tilde{\mathbf{z}}_{r,k} = \mathbf{z}_{r,k} - \mathbf{H}_{r,k} \hat{\mathbf{x}}_{c,k|k} \quad (4.15)$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{c,k|k} + \mathbf{K}_{r,k} \tilde{\mathbf{z}}_{r,k} \quad (4.16)$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{c,k|k} - \mathbf{K}_{r,k} \mathbf{H}_{r,k} \mathbf{P}_{c,k|k}$$

Fig.4.15 presents the system architecture when integrating with a single pseudo-range measurement. It can be extended for multiple pseudo-ranges. Notably, the VIO system is the principle module, while the pseudo-range sensory system is secondary. These two systems are independent; the pseudo-range measurement update cannot intervene in the VIO operation. This property helps to sustain trajectory tracking, even in the case of no communication with any anchor.

The matrix $\mathbf{H}_{r,k} = \left. \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \right|_{\hat{\mathbf{x}}_{k|k-1}}$ describes how the filter states are mapped to the pseudo-range measurement outputs, while it is computed by applying first-order Taylor series approximations to the pseudo-range model. The filter state includes three poses at times k ,

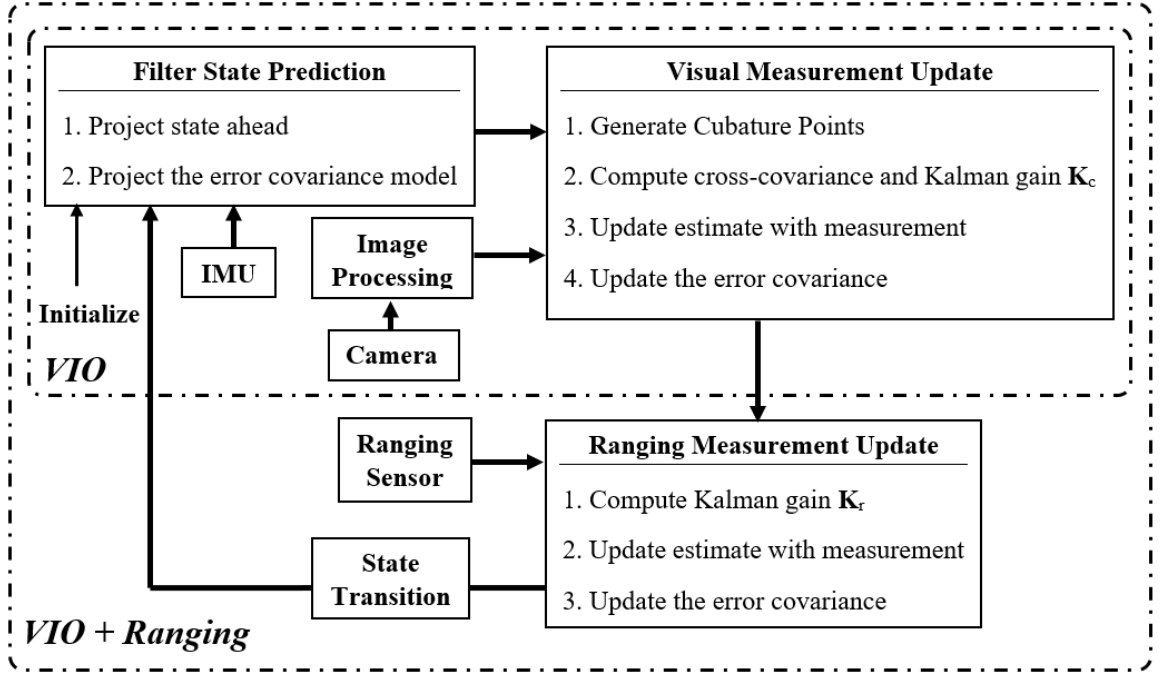


Figure 4.15: System architecture for integrating pseudo-ranging measurement

$k - 1$ and $k - 2$, which are guaranteed to satisfy the constraint of trifocal tensor geometry and epipolar geometry. Consequently, the pseudo-range measurement model will be reconstructed so that it also corrects three poses at each time step.

$$\mathbf{z}_{r,k} = \begin{bmatrix} z_{r,k} & z_{r,k-1} & z_{r,k-2} \end{bmatrix}^T \quad (4.17)$$

The pseudo-range sensor is employed to supplement VIO. It is important to identify and reject pseudo-range measurement outliers before fusing with VIO. Mahalanobis distance d_k of innovation covariance $\mathbf{S}_{r,k}$ and innovation $\tilde{\mathbf{z}}_{r,k}$ are measured to form the validation measurement gate, which defines the region in the pseudo-range measurement space where valid measurements can be found. Any measurement outside that region is considered as an outlier and will not be integrated with VIO. The innovation covariance $\mathbf{S}_{r,k}$ is computed

as in (4.18) before evaluating the validation gate (4.19).

$$\mathbf{S}_{r,k} = \mathbf{H}_{r,k} \mathbf{P}_{c,k|k} \mathbf{H}_{r,k}^T + \mathbf{R}_{r,k} \quad (4.18)$$

$$d_k^2 = \tilde{\mathbf{z}}_{r,k}^T \mathbf{S}_{r,k} \tilde{\mathbf{z}}_{r,k} \leq \gamma_G \quad (4.19)$$

where γ_G is the gate threshold to reject the pseudo-range outliers.

4.4.2 Experimental Validation of Solution 2

The KITTI dataset is also utilized to validate solution 2. In Fig.4.16, multiple anchors are presented as a cyan square marker. The estimation of solution 2 is denoted as VIO+Ranging. Figs.4.16 and 4.17 shows the improvement of estimation accuracy. The use of additional pseudo-range sensors helps to bound the estimation error over long-term operation. Additionally, in these zoom-in images of Fig.4.16, the vehicle is estimated off the road at some specific locations. The system does not consider the issue of obstacle avoidance, which can be solved using a laser-range sensor such as LIDAR.

4.5 Discussion and Conclusions

This section will discuss the pros and cons of the two proposed solutions, which helps to determine the scenario application of each in terms of system accuracy and hardware implementation. Solution 1 implements iterated CKF, which optimizes the latest filter state and covariance during the measurement update. Solution 2 employs pseudo-ranging measurements to bound the estimation errors.

Estimation Accuracy: The experimental results reveal how each solution can improve estimation accuracy. Figure 4.17 evaluates the average RMSE of position estimation for

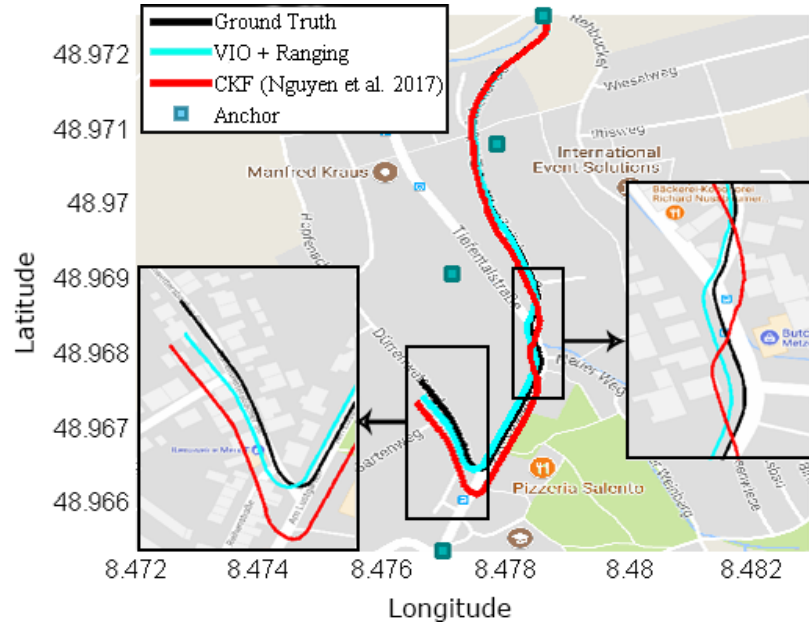


Figure 4.16: Position estimation of dataset 2011_09_30_0034 presented on Google map

each solution. The effectiveness of solution 1 depends on the termination criteria, which decide the number of iterations. Meanwhile, the placement of the pseudo-ranging sensors will influence the number of required ranging corrections to bound the estimation errors, which in turn affects the outcome of solution 2.

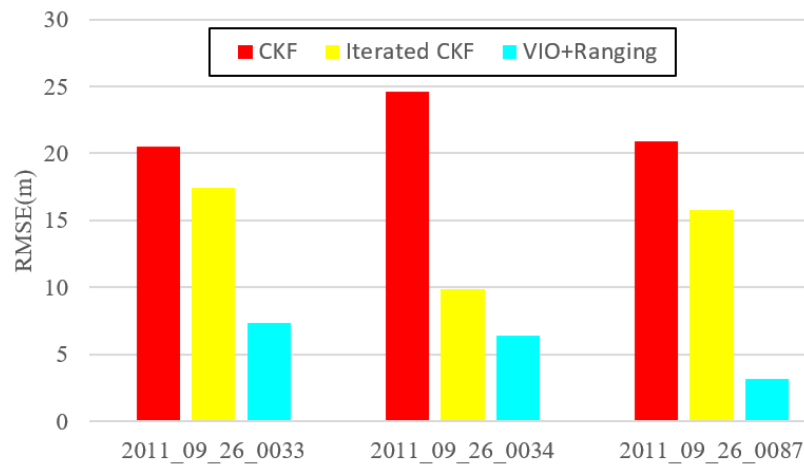


Figure 4.17: RMSE evaluation of position estimation between these solutions

Hardware Implementation: Fig. 4.18 measures the average processing time between these solutions. Both approaches can reduce error at the cost of increasing the processing time. The implementation of the iterated CKF increases the processing time by about 125%, while for solution 2 the cost is reduced by 20%. The iterated CKF consumes more processing time than the VIO+Ranging due to the dimension of the measurement model. Solution 1 executes multiple iterations of the visual measurement update to optimize the estimation, while solution 2 only needs about 1-3 pseudo ranging measurements. The dimensions of the feature-based measurement model is much larger than those of the pseudo-range measurement model. Solution 1 increases the computational cost significantly, which may limit applicability to micro- and nano- robot applications. To improve the computational efficiency, the system can select a small subset of tracked landmark features, which in turn decreases the dimension of the visual measurement model. Alternatively, the implementation of parallel processing can help to speed up the execution [157]. Future work will include further investigation for optimal hardware implementation of iterated CKF.

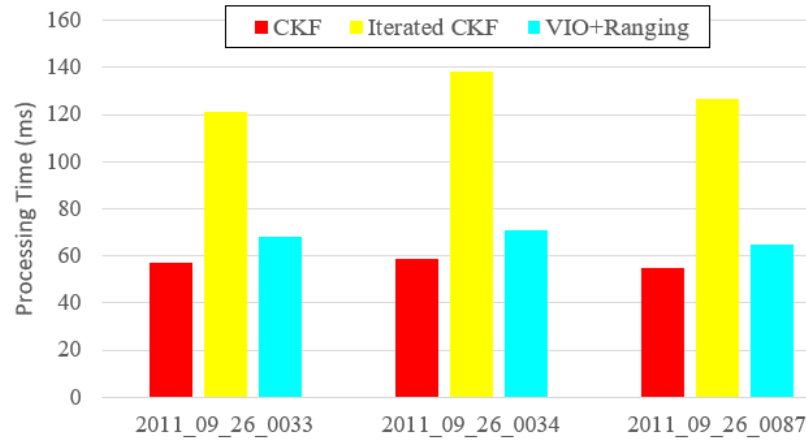


Figure 4.18: Processing time evaluation of these solutions

Although both solutions allow the VIO estimation to operate for a longer duration, the hardware/software requirement of each solution is another consideration for implementation. Solution 1 with MLE-based optimization does not require the installation of advanced optimization library such as Google’s Ceres solver [16] or CasADi [1]. It is feasible to use the same system configuration for implementation, even with hardware constrained platforms. On the other hand, solution 2 requires the setup of multiple anchors along the vehicle trajectory. It cannot be applied to unknown environments.

Overall, both solutions allow the VIO to operate for a longer duration without using a map or executing the optimization of the entire trajectory. A VIO developer will consider these mentioned advantages and disadvantages of each solution, the possible hardware configuration and budget limitation before selecting a solution to upgrade the available VIO system. In general, solution 2 is more suitable for large-scale navigation projects, while solution 1 is preferable for stand-alone self-localization projects.

Chapter 5

Efficient Parallel Processing Solution to Accelerate Visual Inertial Odometry Execution on a Generic Computer

This chapter¹ presents an advanced sensor-fusing algorithm for VIO with parallel processing capability. Conventional systems are inefficient when processing massive amounts of visual data in a serial manner during the filter measurement update. It is impractical to optimize the estimation due to the high computational load and long processing times. As each feature is processed independently in parallel, a novel algorithm is proposed to synthesize individual estimation as well as to maintain the comparative estimation accuracy

¹This chapter is based on the following paper of the author:

- T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "Accelerating Visual Inertial Odometry Using Parallel Processing and Cubature Information Filter", IEEE/ASME Transactions on Mechatronics, (Under Review), 2019

and consistency. Also, the proposed system architecture executes the individual optimization for each feature measurement update. This helps minimize error accumulation over a long-term operation. Experiments use multiple hardware computing platforms and publicly available datasets for validation. This chapter also redefines the error correction and builds the computation on Matrix Lie groups with invariant Kalman filtering, which helps to achieve the consistency during operation.

5.1 Introduction

Efficiency via parallel computing. VIO demands heavy computation to process the camera image at each time instant, and this makes hardware implementation quite challenging and non-trivial. It is inefficient to process massive amounts of visual data in a serial manner during the filter measurement update. As a result, it is very difficult to optimize the estimation with minimum processing time. Parallel processing is an alternative answer to implement the local optimization of the individual feature as well as speed up the execution. There are many strategies to enhance parallel processing for VIO application: FPGAs [157], GPUs [82, 126, 151] and ASICs [131]. However, the development of these specialized hardware computing platforms for perception is an expensive and time-consuming process. Also, the resulting hardware is difficult to upgrade [28]. Before making an investment in these expensive solutions, it is useful to verify the VIO parallel computing on a generic computer. At the moment, multi-core CPU is popularly available in all types of computers. This solution is selected to enhance parallel processing and accelerate the execution. In reality, most projects have been utilizing multi-core CPU to develop their VIO algorithm with minimum consideration on how the hardware distributes the computation over multi-

ple cores. Alternatively, the computational task on each core is assigned during the visual measurement update to accelerate the VIO computation.

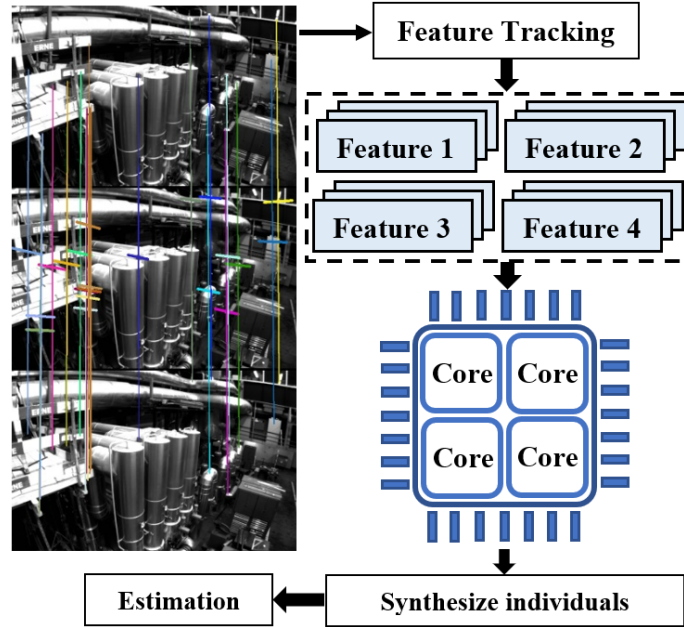


Figure 5.1: Parallel processing for each feature

Individual processing each visual feature. Not all features have the same contribution to the estimation accuracy. Motion blur and various texture gradients can reduce a feature’s positive impact on the filter update [114]. Some researches [28, 112, 114, 159] have selected to evaluate each feature independently during the measurement update step. For example, Peretroukhin et al. [112, 114] have built a visual measurement noise model to predict the noise covariance parameter for each feature at each time instant. Furthermore, Carlone et al. [28] have developed a visual attention mechanism that measures the contribution of each feature to the filter estimation, and then selects a suitable subset of landmark features for the VIO operation. Consequently, the number of features is reduced positively to apply for hardware-constrained computing systems but maximize the localization accuracy.

However, this strategy needs to distribute the computation such that the processing of many small-scale visual measurement models does not increase the processing time significantly. In reality, most VIO algorithms make a strong assumption about all visual features having equal priority to the filter. The system constructs all of them into a single composite group measurement model to compute in the scheme of Kalman based filtering. This strategy results in a higher-dimensional matrix of measurement model and innovation covariance, which are not computationally efficient [31, 33, 103, 109]. In this chapter, individual features are assumed to have a different contribution to the motion estimation, which allows a single core to process one feature measurements independently.

Drifts over long-term operation. As mentioned in the previous chapter, the local optimization within the sliding window can help to address the error accumulation issue over long-term operation. Optimally tuning parameters and terminating the optimization process will decide the effectiveness of the technique. The implementation does not require the installation of any specialized hardware platform and some advanced optimization software library. The employment of local optimization raises two interesting questions: how to perform the local optimization for individual feature? And how to distribute the massive computation of the optimization process on multi-core CPU such that the increase in processing time is not significant?

This chapter describes research related to distributing the VIO computation on a multi-core CPU. The computational task is strictly assigned for each core to process each feature independently at each time instant (Fig.5.1). The computational tasks include the visual measurement update and local optimization process for a single feature. The synthesis of all individual feature processing is designed to exploit the computationally efficient benefits of CIF. To summarize, this chapter makes the following contributions.

- Firstly, parallel processing for VIO computation is enhanced using CIF and multi-core CPU. It speeds up the estimation significantly while maintaining a comparative estimation error. This is proposed to be the first deployment of parallel processing for VIO applications on a multi-core CPU solution.
- Secondly, optimization for individual feature measurement update is deployed to minimize the error accumulation over long-term operation. Multiple corrections on single measurement are conducted to produce optimal estimation. The implementation does not require the installation of advanced optimization software libraries such as Google's Ceres solver [16].
- Lastly, an efficient and easy method to accelerate the VIO execution without the installation of specialized hardware computing platforms is introduced. As this method can be expended to other advanced hardware platforms, this is considered as the middle step to validate the VIO parallel computing on a generic computer before investing in specialized hardware.

The remainder of the chapter is organized as follows. The next section reviews some related researches. Section 5.2 presents system coordinates, matrix Lie groups and the problem formulation. The proposed filtering algorithm has two main parts: prediction and update, which are described in section 5.3 and 5.4, respectively. Then, the system architecture is proposed in section 5.5. Section 5.6 describes some experiments for evaluating the proposed algorithm. Lastly, section 5.7 provides conclusions.

5.2 Preliminaries

5.2.1 Matrix Lie Groups

A matrix Lie group G ($G \subset \mathbb{R}^{d \times d}$) is a subset of square invertible matrices possessing the properties: $\mathbf{I} \in G$, $\forall \chi \in G$, $\chi^{-1} \in G$, $\forall \chi_1, \chi_2 \in G$, $\chi_1 \chi_2 \in G$. The group forms a smooth manifold. A Lie algebra \mathfrak{g} is a tangent space of the group's manifold at the identity matrix \mathbf{I} . In our case, a rotation matrix \mathbf{R} has the essential properties of the special orthogonal group $\text{SO}(3)$: $\mathbf{R} \in \text{SO}(3) \doteq \{\mathbf{R} \in \mathbb{R}^{3 \times 3} : \mathbf{R}^T \mathbf{R} = \mathbf{R} \mathbf{R}^T = \mathbf{I}, \det(\mathbf{R}) = 1\}$. The Lie algebra $\mathfrak{so}(3)$ coincides with the space of skew symmetric matrices. Every vector $\boldsymbol{\omega} \in \mathbb{R}^3$ can produce its skew symmetric matrix form $\boldsymbol{\phi}^\wedge$ using the hat operator $(\cdot)^\wedge$ (5.1). In reverse, a skew symmetric matrix can produce its vector \mathbb{R}^3 form with $(\cdot)^\vee$ vee operator. Special Euclidean Group $\text{SE}(3)$ describes the group of 3D rigid motion, $\text{SE}(3) = \{(\mathbf{R}, \mathbf{p}) : \mathbf{R} \in \text{SO}(3), \mathbf{p} \in \mathbb{R}^3\}$. The group operation is $\mathbf{T}_1 \mathbf{T}_2 = (\mathbf{R}_1 \mathbf{R}_2, \mathbf{R}_1 \mathbf{p}_2 + \mathbf{p}_1)$ and $\mathbf{T}_1^{-1} = (\mathbf{R}_1^T, -\mathbf{R}_1^T \mathbf{p}_1)$.

$$S(\boldsymbol{\phi}) = [\boldsymbol{\phi}]^\wedge = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix}^\wedge = \begin{bmatrix} 0 & -\phi_3 & \phi_2 \\ \phi_3 & 0 & -\phi_1 \\ -\phi_2 & \phi_1 & 0 \end{bmatrix} \in \mathfrak{so}(3) \quad (5.1)$$

The exponential ($\exp(\cdot)$) and logarithm ($\log(\cdot)$) operations can be used to link (or map) between a Lie group and its associated Lie algebra (5.2) [19, 148].

$$\begin{aligned} \mathbb{R}^3 &\rightarrow \text{SO}(3) & : \boldsymbol{\phi} &\rightarrow \exp(\boldsymbol{\phi}^\wedge) \\ \text{SO}(3) &\rightarrow \mathbb{R}^3 & : \mathbf{R} &\rightarrow \log(\mathbf{R})^\vee \end{aligned} \quad (5.2)$$

The Lie group action is defined in the right group: $\chi \sim \mathcal{N}_R(\hat{\chi}, \mathbf{P})$, $\xi \sim \mathcal{N}(\mathbf{0}, \mathbf{P})$, $\chi = \exp(\xi) \hat{\chi}$. Assuming χ is a random variable of the Lie group ($\chi \in G$), the true value χ consists of the noise-free nominal value $\hat{\chi}$ and the error value (perturbation) ξ , where ξ follows

the Gaussian distribution with its covariance \mathbf{P} . Notably, the probability density function is built on vector space \mathbb{R}^d such that $\mathcal{N}_L(.,.)$ and $\mathcal{N}_R(.,.)$ are not Gaussian distributions.

5.2.2 Problem Formulation

The coordinate system is defined as in the previous chapter with $\{G\}$ is a global frame; $\{I\}$ is an IMU frame; $\{C\}$ is a camera frame. The system attempts to track the transformation of $\{I\}$ with respect to $\{G\}$. The evolution of the system state sequence χ_k at discrete time k operates by: $\chi_k = \mathbf{f}(\chi_{k-1}, \mathbf{u}_k, \mathbf{w}_k)$, where \mathbf{f} is the nonlinear function, \mathbf{u}_k is the input and \mathbf{w}_k is the process noise following zero-mean white Gaussian model with covariance \mathbf{Q}_k . A measurement $\mathbf{z}_k \in \mathbb{R}^{n_z}$ arrives with $\mathbf{z}_k = \mathbf{h}(\chi_k) + \mathbf{V}_n$, where \mathbf{V}_n is the measurement noise following zero-mean white Gaussian noise with covariance \mathbf{R}_k .

The Bayesian filtering sequentially computes the Probability Density Function (**PDF**) of the filtering's posterior state $\mathbb{P}(\chi_k | \mathbf{z}_{1:k})$ with $\mathbf{z}_{1:k} = \{\mathbf{z}_1, \dots, \mathbf{z}_k\}$. This can be achieved using two step processes (prediction and measurement update), when the PDF of the filtering posterior state at the previous step $k-1$ is available. The prediction step results in the predictive posterior state PDF: $\mathbb{P}(\chi_k | \mathbf{z}_{1:k-1}) = \int \mathbb{P}(\chi_k | \chi_{k-1}) \mathbb{P}(\chi_{k-1} | \mathbf{z}_{1:k-1}) d\chi_{k-1}$. The new measurement is used to update the predictive posterior state PDF through Bayes rules:

$$\mathbb{P}(\chi_k | \mathbf{z}_{1:k}) = \frac{\mathbb{P}(\chi_k | \mathbf{z}_{1:k-1}) \mathbb{P}(\mathbf{z}_k | \chi_k)}{\int \mathbb{P}(\chi_k | \mathbf{z}_{1:k-1}) \mathbb{P}(\mathbf{z}_k | \chi_k) d\chi_k} \quad (5.3)$$

The measurement vector is an uncorrelated high-dimensional vector composed of multiple low-dimensional vectors (i.e. $\mathbf{z}_k = [\mathbf{z}_{1,k}^T, \mathbf{z}_{2,k}^T, \dots, \mathbf{z}_{M,k}^T]$). The covariance matrix can be applied similarly: $\mathbf{R}_k = \text{diag}(\mathbf{R}_{1,k}, \mathbf{R}_{2,k}, \dots, \mathbf{R}_{M,k})$. A hardware platform with D processors is employed to enhance parallel processing. The measurement vector can be decomposed into D measurement vectors, which are processed in parallel: $\{\mathbf{z}_{k,d}^D\}_{d=1}$, with $\mathbf{z}_k = \bigcup_{d=1}^D \mathbf{z}_{k,d}$ and

$\mathbf{z}_{k,i} \cap \mathbf{z}_{k,j} = \emptyset \forall i \neq j$. The (5.3) is updated accordingly:

$$\mathbb{P}(\chi_k | \mathbf{z}_{1:k}) = \frac{\mathbb{P}(\chi_k | \mathbf{z}_{1:k-1}) \prod_{d=1}^D \mathbb{P}(\mathbf{z}_{k,d} | \chi_k)}{\int \mathbb{P}(\chi_k | \mathbf{z}_{1:k-1}) \prod_{d=1}^D \mathbb{P}(\mathbf{z}_{k,d} | \chi_k) d\chi_k} \quad (5.4)$$

where $\mathbb{P}(\mathbf{z}_{k,d} | \chi_k)$ is the local measurement likelihood for each processor d . Under Gaussian approximation, the Bayesian filter's functional recursion will be reduced to an algebraic recursion operating on means $\hat{\chi}_k$ and covariances \mathbf{P}_k of various conditional densities encountered in the time and measurement updates within the Kalman filtering framework [18]. The estimation is expected to be consistent as a result of the invariant property. Let the system perform on a matrix Lie group G while \mathbf{f} is a smooth map on the group. The system will be redefined to achieve the invariant property as introduced in [20] and repeated below for the reader's convenience.

Theorem 1. *The smooth map on the group \mathbf{f} satisfying equation (5.5) \Leftrightarrow There exists a map \mathbf{g} such that $\mathbf{f}(\mathbf{v}, \mathbf{u})^{-1} \mathbf{f}(\chi, \mathbf{u}) = \mathbf{g}(\mathbf{v}^{-1} \chi, \mathbf{u}) \forall \chi, \mathbf{v}, \mathbf{u}$. Additionally, for each $\mathbf{u} \in \mathbb{R}^m$, there exists $\mathbf{F} \in \mathbb{R}^{d \times d}$ such that $\forall \xi \in \mathbb{R}^d$, $\mathbf{g}(\cdot, \mathbf{u}) = \exp(\mathbf{F}\xi)$ is wholly encoded in a simple matrix \mathbf{F} .*

$$\mathbf{f}(\chi \mathbf{v}, \mathbf{u}) = \mathbf{f}(\chi, \mathbf{u}) \mathbf{f}^{-1}(\mathbf{I}, \mathbf{u}) \mathbf{f}(\mathbf{v}, \mathbf{u}) \quad (5.5)$$

The system with this property is a group affine system: $\chi_n \in G : \chi_k = \mathbf{f}(\chi_{k-1}, \mathbf{u}_k) \exp(\mathbf{w}_k)$. Similarly, the invariant Kalman filter also produces an estimation for mean $\hat{\chi}_{k|k} \in G$ and covariance $\mathbf{P}_{k|k} \in \mathbb{R}^{d \times d}$. The right-invariant error is defined as (5.6) to compute the associated covariance. This error is invariant for right multiplications [19]. $(\chi, \hat{\chi}) \mapsto (\Gamma \chi, \Gamma \hat{\chi})$ for any group element $\Gamma \in G$. This error will be used for measuring the discrepancy between the true state χ and the estimate $\hat{\chi}$ in Lie group context. That representation will replace the usual linear error $\hat{\chi} - \chi$ in a conventional Kalman filter, which is not an element of G .

When \mathbf{f} satisfies the invariant condition (5.5), theorem 1 yields (5.7).

$$\eta_{k-1|k-1} = \hat{\chi}_{k-1|k-1} \chi_{k-1}^{-1}, \quad \eta_{k|k-1} = \hat{\chi}_{k|k-1} \chi_k^{-1} \quad (5.6)$$

$$\begin{aligned} \eta_{k|k-1} &= \mathbf{f}(\hat{\chi}_{k-1|k-1}, \mathbf{u}_k) (\mathbf{f}(\chi_{k-1}, \mathbf{u}_k) \exp(\mathbf{w}_k))^{-1} \\ &= \exp(-\mathbf{w}_k) \mathbf{g}(\eta_{k-1|k-1}, \mathbf{u}_k) \end{aligned} \quad (5.7)$$

The conventional Kalman filter is used to linearize the error through a first-order Taylor expansion of the nonlinear functions \mathbf{f} and \mathbf{h} at the estimate $\hat{\chi}$. However, the estimation error (5.6) will be an element of G and a square matrix. When $\hat{\chi}$ and χ are closed, the invariant error η is close to the identity matrix \mathbf{I} . The Lie exponential map can provide a bijection between a neighborhood of \mathbb{R}^d and a neighborhood of \mathbf{I} . Consequently, the estimation error can be locally approximated by an element of \mathbb{R}^d [20]. It results in $\exp(\xi_{k|k-1}) = \eta_{k|k-1} = \hat{\chi}_{k|k-1} \chi_k^{-1}$, or $\chi_k = \exp(\xi_{k|k-1}) \hat{\chi}_{k|k-1}$. It also proves that if $\mathbb{P}(\chi_{k-1} | \mathbf{u}_{1:k-1}, \mathbf{Y}_{1:k-1}) \simeq \mathcal{N}_R(\hat{\chi}_{k-1|k-1}, \mathbf{P}_{k-1|k-1})$ then the propagated distribution is also approximated as $\mathbb{P}(\chi_k | \mathbf{u}_{1:k}, \mathbf{Y}_{1:k}) \simeq \mathcal{N}_R(\hat{\chi}_{k|k-1}, \mathbf{P}_{k|k-1})$, where $\mathbf{Q}_k = \text{Cov}(\mathbf{w}_k) = \text{Cov}(-\mathbf{w}_k)$.

In the update step, the $\mathbf{z}_k \in \mathbb{R}^p$ is obtained as $\mathbf{z}_k = \mathbf{Y}_k - \mathbf{h}(\hat{\chi}_{k|k-1}) + \mathbf{V}_k = \mathbf{h}(\chi) - \mathbf{h}(\hat{\chi}_{k|k-1}) + \mathbf{V}_k$. Substituting the defined invariant state estimation, $\mathbf{z}_k = \mathbf{h}(\exp(\xi_{k|k-1}) \hat{\chi}_{k|k-1}) - \mathbf{h}(\hat{\chi}_{k|k-1}) + \mathbf{V}_k$ is obtained. Assuming $\xi_{k|k-1}$ is small and $\exp(\mathbf{0}) = \mathbf{I}$, a first-order Taylor expansion in $\xi \in \mathbb{R}^d$ arbitrary allows defining \mathbf{H}_k as (5.8).

$$\mathbf{h}(\exp(\xi_{k|k-1}) \hat{\chi}_{k|k-1}) - \mathbf{h}(\hat{\chi}_{k|k-1}) + \mathbf{V}_k = \mathbf{H}_k \xi + O(\|\xi\|^2) \quad (5.8)$$

As a result, the invariant Kalman filter can be applied similarly to the conventional Kalman filter.

5.3 Filter State Formation and Prediction

The filter state χ_k consists of the IMU state and the last IMU poses: $\chi_k = (\mathbf{x}_k, \mathbf{C}_1, \dots, \mathbf{C}_i)$.

The IMU state is described by $\mathbf{x}_k = ({}^G\mathbf{p}_I, {}^G\mathbf{R}_I, {}^G\mathbf{v}_I, \mathbf{b}_a, \mathbf{b}_g)$. The previous IMU pose is denoted as $\mathbf{C}_i = ({}^G\mathbf{p}_{I_i}, {}^G\mathbf{R}_{I_i})$ which is associated with image I_i . The Lie group is intrinsically employed to represent the uncertainty of the IMU state: $\mathbf{x}_k = \hat{\mathbf{x}}_k \oplus \xi_k$ with $\xi_k \sim \mathcal{N}_R(\mathbf{0}, \mathbf{P})$.

The symbol \oplus denotes the retraction in differentiable geometry as in (5.9) and (5.10).

Consequently, the filter true state will consist of the nominal state $\hat{\chi}_k = (\hat{\mathbf{x}}_k, \hat{\mathbf{C}}_1, \dots, \hat{\mathbf{C}}_i)$

and the error state $\tilde{\chi} = (\xi_k, \xi_{\mathbf{C}_1}, \dots, \xi_{\mathbf{C}_i})$, which is expressed through equation $\hat{\chi} \oplus \tilde{\chi} = (\hat{\mathbf{x}}_k \oplus \xi_k, \hat{\mathbf{C}}_1 \oplus \xi_{\mathbf{C}_1}, \dots, \hat{\mathbf{C}}_i \oplus \xi_{\mathbf{C}_i})$ with $\xi_{\mathbf{C}_i} = (\xi_{\mathbf{p}_i}, \xi_{\mathbf{R}_i})$.

$$\hat{\mathbf{x}}_k \oplus \xi_k = \begin{pmatrix} \exp(\xi_{\mathbf{R}}) {}^G\hat{\mathbf{p}}_I + \mathbf{J}_r(-\xi_{\mathbf{R}})\xi_{\mathbf{p}} \\ \exp(\xi_{\mathbf{R}}) {}^G\hat{\mathbf{R}}_I \\ \exp(\xi_{\mathbf{R}}) {}^G\hat{\mathbf{v}}_I + \mathbf{J}_r(-\xi_{\mathbf{R}})\xi_{\mathbf{v}} \\ \hat{\mathbf{b}}_a + \xi_{\mathbf{b}_a} \\ \hat{\mathbf{b}}_g + \xi_{\mathbf{b}_g} \end{pmatrix} \quad (5.9)$$

$$\hat{\mathbf{C}}_i \oplus \xi_{\mathbf{C}_i} = \begin{pmatrix} \exp(\xi_{\mathbf{R}_i}) {}^G\hat{\mathbf{p}}_{I_i} + \mathbf{J}_r(-\xi_{\mathbf{R}_i})\xi_{\mathbf{p}_i} \\ \exp(\xi_{\mathbf{R}_i}) {}^G\hat{\mathbf{R}}_{I_i} \end{pmatrix} \quad (5.10)$$

Prediction is used to infer the motion from the IMU measurements and the kinematic model. The state at the time $k + \Delta t$ can be computed by numerical integration. The Lie group version of Runge-Kutta numerical integration is applied, which enables the compositions of exponentials or flows instead of commutator corrections, as well as preserves the convergence at a higher-order numerical integration [30]. Setting $k_i = \Delta t \mathbf{f}_{\text{IMU}}(Y_i)$ for all i ,

the integration is computed:

$$\begin{aligned}
Y_1 &= y_0; & Y_2 &= \exp\left(\frac{1}{2}k_1\right)y_0 \\
Y_3 &= \exp\left(\frac{1}{2}k_2\right)y_0; & Y_4 &= \exp\left(k_3 - \frac{1}{2}k_1\right)Y_2 \\
y_{\frac{1}{2}} &= \exp\left(\frac{1}{2}(3k_1 + 2k_2 + 2k_3 - k_4)\right)y_0 \\
y_1 &= \exp\left(\frac{1}{2}(-k_1 + 2k_2 + 2k_3 + 3k_4)\right)y_{\frac{1}{2}}
\end{aligned} \tag{5.11}$$

The filter error state will follow the linearized error-state propagation model $\dot{\xi} = \mathbf{F}\xi + \mathbf{G}\mathbf{n}_{\text{IMU}} + o(\|\xi\| \|\mathbf{n}_{\text{IMU}}\|)$ with these time-varying Jacobian matrices \mathbf{F} and \mathbf{G} . The state transition matrix $\Phi := \Phi(k, k+1)$ at the time $k+1$ follows the ODE: $\frac{d}{dt}\Phi(k, k+1) = \mathbf{F}(k)\Phi(k, k+1)$ with condition $\Phi(k, k) = \mathbf{I}$ at the time k . More information about the matrix structure of \mathbf{F} , \mathbf{G} , Φ , \mathbf{Q}_c and \mathbf{Q}_d can be located in [148]. The discrete-time system noise covariance matrix \mathbf{Q}_d can be constructed from the continuous-time system noise covariance matrix $\mathbf{Q}_c = \text{diag}(\sigma_g^2, \sigma_a^2, \sigma_{wg}^2, \sigma_{wa}^2)$ using \mathbf{Q}_d (5.12). Finally, the state covariance matrix $\mathbf{P}_{k-1|k-1}$ can be propagated as in (5.13).

$$\mathbf{Q}_d = \int_{\Delta t} \Phi_d(\tau) \mathbf{G}(\tau) \mathbf{Q}_c \mathbf{G}^T(\tau) \Phi_d^T(\tau) d\tau \tag{5.12}$$

$$\mathbf{P}_{k|k-1} = \Phi \mathbf{P}_{k-1|k-1} \Phi^T + \mathbf{Q}_d \tag{5.13}$$

5.4 Filter Measurement Update

Deterministic sampling based filters have been proposed as a better solution to handle the highly nonlinear model. This chapter employs CKF, which generates a weight set of symmetrical points to address the nonlinearity. According to [18], the Bayesian filter solution is reduced to approximating multi-dimensional integrals, for which the integrands are of the

form *nonlinear function* \times *Gaussian*. The heart of CKF utilizes the cubature rule to find these multi-dimensional integrals. We also have $\mathbb{P}(\chi_k | \mathbf{u}_{1:k}, \mathbf{Y}_{1:k}) \simeq \mathcal{N}_R(\hat{\chi}_{k|k-1}, \mathbf{P}_{k|k-1})$ and $\chi_k = \exp(\xi_{k|k-1}) \hat{\chi}_{k|k-1}$ in the invariant Kalman filter. The cubature rule to approximate a n -dimensional Gaussian weighted integral is redefined in the right group action as (5.14).

$$I_{\mathcal{N}_R}(\mathbf{f}) = \int_{\mathbb{R}^n} \mathbf{f}(\mathbf{x}) \mathcal{N}_R(\chi; \hat{\chi}, \mathbf{P}) d\mathbf{x} \simeq \sum_{s=1}^{2n} \frac{1}{2n} \mathbf{f}(\exp(\mathbf{S}\alpha_s) \hat{\chi}) \quad (5.14)$$

where the covariance matrix \mathbf{P}_k has its square-root factor \mathbf{S}_k with $\mathbf{P}_k = \mathbf{S}_k \mathbf{S}_k^T$. If a system has n state variables, the third-order Cubature Kalman Filter will select $2n$ cubature points. The cubature point set $\{\alpha_s\}$ is generated as in (5.15).

$$\alpha_s = \begin{cases} \sqrt{n} e_s & s = 1, 2, \dots, n \\ -\sqrt{n} e_{s-n} & s = n+1, n+2, \dots, 2n \end{cases} \quad (5.15)$$

Where $e_s \in \mathbb{R}^n$ represents the s^{th} elementary column vector. Applied to the VIO system, these cubature points generated by using (5.15) are evaluated by (5.16) with $\xi_{k|k-1} = \mathbf{0}_{27 \times 1}$. The propagated cubature points are evaluated in (5.17) before estimating the predicted measurement (5.18).

$$\xi_{s,k|k-1} = \mathbf{S}_{k|k-1} \alpha_s + \xi_{k|k-1} \quad (5.16)$$

$$\mathbf{Z}_{s,k|k-1} = \mathbf{h}(\exp(\xi_{s,k|k-1}) \hat{\chi}_{k|k-1}, \{\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3\}) \quad (5.17)$$

$$\hat{\mathbf{z}}_{k|k-1} = \sum_{s=1}^{2n} \frac{1}{2n} \mathbf{Z}_{s,k|k-1}; \quad (5.18)$$

Then, the posterior state and the associated covariance matrix are corrected with the actual measurement \mathbf{z}_k and the visual measurement noise \mathbf{R}_c as (5.19), (5.20) and (5.21).

$$\mathbf{K}_k = \mathbf{P}_{xz} (\mathbf{P}_{zz} + \mathbf{R}_c)^{-1} \quad (5.19)$$

$$\hat{\chi}_{k|k} = \hat{\chi}_{k|k-1} \oplus (\mathbf{K}_k (\mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1})) \quad (5.20)$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{P}_{zz} \mathbf{K}_k^T \quad (5.21)$$

However, (5.19), (5.20) and (5.21) are not computationally efficient for parallel processing, especially with a large number of feature measurements [31, 32, 57]. The filter is transferred to the information domain by taking the inverse of covariance (5.22). The filter will measure information about the states of interest rather than the direct mean and its associated covariance.

$$\mathbf{P}_{k|k-1} = \mathbf{Y}_{k|k-1} \setminus \mathbf{I} \quad (5.22)$$

$$\hat{\boldsymbol{\lambda}}_{k|k-1} = \mathbf{Y}_{k|k-1} \setminus \hat{\mathbf{y}}_{k|k-1} \quad (5.23)$$

$$\mathbf{Y}_k = \mathbf{H}_k^T \mathbf{R}_k^{-1} \mathbf{H}_k \quad (5.24)$$

$$\mathbf{y}_k = \mathbf{H}_k^T \mathbf{R}_k^{-1} (\mathbf{z}_k - \hat{\mathbf{z}}_k^j + \mathbf{H}_k^T \hat{\mathbf{x}}_{k|k-1})$$

In the structure of the extended information filter [32, 57], the information state contribution \mathbf{i}_k and its associated information matrix \mathbf{Y}_k are calculated as (5.24). The innovation covariance matrix and the cross covariance matrix can be approximated as $\mathbf{P}_{zz} = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T$ and $\mathbf{P}_{xz} = \mathbf{P}_{k|k-1} \mathbf{H}_k^T$ [32]. This results in (5.25) and spares the computation of the covariance matrix. We present equation (5.25) in terms of parallel processing for each feature f . As the filtering operates on matrix Lie groups, the correction of \mathbf{y}_k and \mathbf{Y}_k (5.26) is necessary in the right Jacobian of G [31].

$$\mathbf{Y}_{k,f}^- = \mathbf{Y}_{k|k-1} \mathbf{P}_{xz,f} \mathbf{R}_k^{-1} \mathbf{P}_{xz,f}^T \mathbf{Y}_{k|k-1}^T \quad (5.25)$$

$$\mathbf{y}_{k,f}^- = \mathbf{Y}_{k|k-1} \mathbf{P}_{xz,f} \mathbf{R}_k^{-1} (\mathbf{z}_{k,f} - \hat{\mathbf{z}}_{k,f}^j + \mathbf{P}_{xz,f}^T \mathbf{Y}_{k|k-1}^T \hat{\mathbf{x}}_{k|k-1})$$

$$\begin{aligned} \mathbf{Y}_{k,f} &= \Psi \left((\mathbf{Y}_{k,f}^-)^{-1} \mathbf{y}_{k,f}^- \right)^{-T} \mathbf{Y}_{k,f}^- \Psi \left((\mathbf{Y}_{k,f}^-)^{-1} \mathbf{y}_{k,f}^- \right)^{-1} \\ \mathbf{y}_{k,f} &= \mathbf{Y}_{k,f} (\log(\exp(\mathbf{Y}_{k-1}^{-1} \mathbf{y}_k)^\wedge)) \end{aligned} \quad (5.26)$$

$$\exp(\xi_{k|k-1}) \exp((\mathbf{Y}_{k,f}^-)^{-1} \mathbf{y}_{k,f}^-)^\vee$$

A single feature f is observed from a set of camera poses. The feature position in the camera frame ${}^{C_i}\mathbf{p}_f = \begin{bmatrix} {}^{C_i}X_f & {}^{C_i}Y_f & {}^{C_i}Z_f \end{bmatrix}^T$ is computed from the 3D feature position ${}^G\mathbf{p}_f$ as (5.27). Assuming \mathbf{V}_i^f is the image noise vector with covariance \mathbf{R}_c , the measurement model is presented:

$${}^{C_i}\mathbf{p}_f = {}^{C_i}\mathbf{R}_G ({}^G\mathbf{p}_f - {}^G\mathbf{p}_{C_i}) \quad (5.27)$$

$$\mathbf{z}_i^f = \frac{1}{{}^{C_i}Z_f} \begin{bmatrix} {}^{C_i}X_f \\ {}^{C_i}Y_f \end{bmatrix} + \mathbf{V}_i^f \quad (5.28)$$

5.5 System Architecture

5.5.1 Distribute Computation on Multiple Cores

Figure 5.2 illustrates the architecture of a quad-core processor having four physical cores for computation. It also includes cache memory at three levels: L1, L2 and L3. It is a high-speed Static Random Access Memory (**SRAM**) that CPU can access faster than a regular Random Access Memory (**RAM**). Although a cache helps to reduce the average cost of time and energy to access data from the main memory, its small size may limit the computational acceleration. The deployment of parallel processing is only satisfied when the time spent on computation significantly exceeds the time spent on transferring data between the CPU and memory. This condition results in the minimum number of features to distribute the computation effectively.

Figure 5.3 describes the proposed VIO system architecture. The *image processing* block performs feature detection and matching, which provides reliable feature measurements to correct the predicted filter state. When a new camera image arrives, the filter is transferred to information form using (5.22) and (5.23). Then, the parallel processing

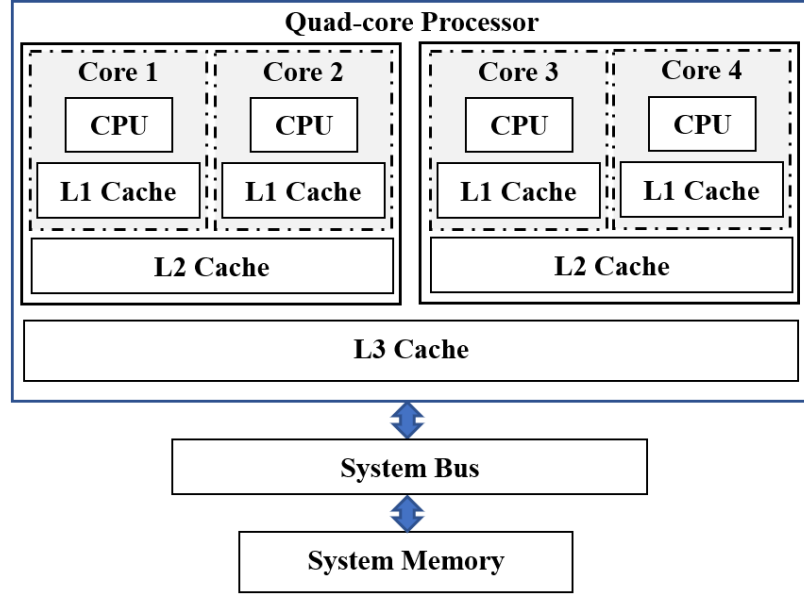


Figure 5.2: The architecture of a multi-core CPU

initialization will prepare multiple measurement models and multiple available cores to evaluate the contribution of each feature to the motion estimation (i.e., $\mathbf{y}_{k,f}$ and $\mathbf{Y}_{k,f}$). Assuming the system is tracking n_f features, the evaluation of the contribution of each feature f is executed in parallel on n_{core} cores. Each execution is independent, while n_{core} is much less than n_f because of using the common multi-core computer. After individual evaluation, Eq.(5.29) is applied to synthesize at an additive form. This synthesis is beneficial when each feature requires a different processing time for evaluation and optimization. According to the common structure of the information filter [31, 32, 57], the system also requires the transferring back to state form for the prediction step.

$$\hat{\mathbf{y}}_{k|k} = \hat{\mathbf{y}}_{k|k-1} + \sum_{f=1}^{n_f} \mathbf{y}_{k,f}; \quad \mathbf{Y}_{k|k} = \mathbf{Y}_{k|k-1} + \sum_{f=1}^{n_f} \mathbf{Y}_{k,f} \quad (5.29)$$

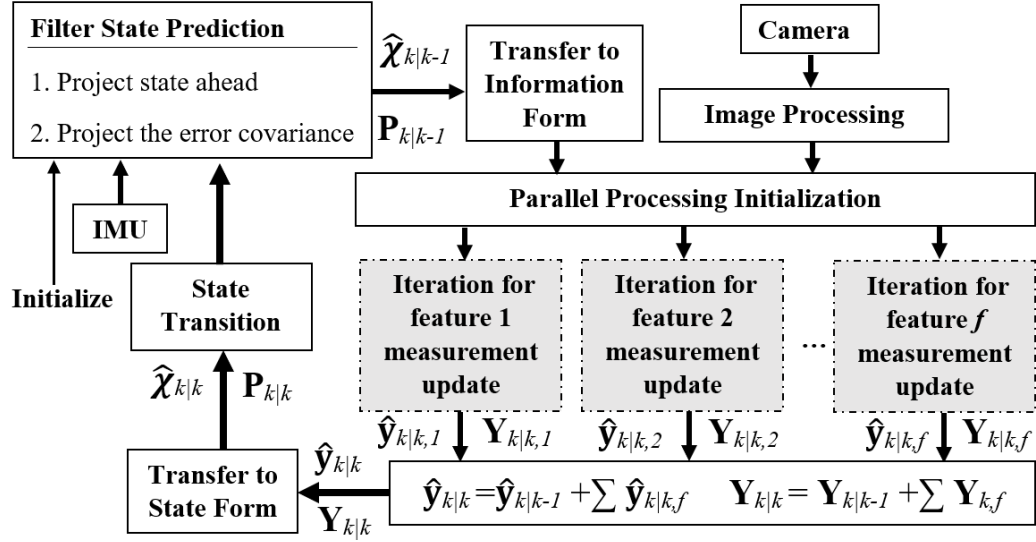


Figure 5.3: The proposed VIO system architecture where the gray blocks are executed in parallel on a multi-core CPU

5.5.2 Optimization during Measurement Update

To address the error accumulation issue over long-term operation, the iteration procedure is deployed for the visual measurement update, which optimizes the estimation. As distributing the computation on multiple cores, the optimization process is also applied for each individual feature measurement. In the optimization process of a single feature f , the estimated state $\hat{\mathbf{x}}_{k|k,f}$ is closer to the filter true state than the predicted state $\hat{\mathbf{x}}_{k|k-1,f}$. In the iteration, the estimated state $\hat{\mathbf{x}}_{k|k,f}^j$ at j^{th} iteration produces a better approximation to the filter true state than the estimated state $\hat{\mathbf{x}}_{k|k,f}^{j-1}$ at $(j-1)^{th}$ iteration.

Similar to the previous chapter, the optimization during the visual measurement update is built on maximum likelihood estimate. Assuming $\Delta \mathbf{z}_{k,f}^j = \mathbf{z}_{k,f} - \hat{\mathbf{z}}_{k,f}^j$, $\mathbf{x}_{k|k,f} \sim \mathcal{N}(\mathbf{x}_{k|k,f}^j, \mathbf{P}_{k|k,f}^j)$ and $\mathbf{z}_{k,f} \sim (\hat{\mathbf{z}}_{k,f}^j, \mathbf{R}_c)$, the cost function is determined in the state domain (5.30). Since $(\mathbf{Y}_{k,f}^{(j-1)})^{-1} = \mathbf{S}_{k,f}^{(j-1)}(\mathbf{S}_{k,f}^{(j-1)})^T$, the cost function can be rewritten in

information domain as (5.31). During the optimization process, the inequality condition $J(\hat{\chi}_{k|k,f}^j) < J(\hat{\chi}_{k|k,f}^{j-1})$ is always satisfied, where $J(\hat{\chi}_{k|k,f}^j)$ is closer to the maximum likelihood surface than $J(\hat{\chi}_{k|k,f}^{j-1})$. $\hat{\chi}_{k|k,f}^j$ is a more accurate approximation of MLE than $\hat{\chi}_{k|k,f}^{j-1}$. The inequality is used for terminating the iteration.

$$J(\hat{\chi}_{k|k,f}^j) = (\Delta\hat{\chi}_{k|k,f}^j)^T (\mathbf{S}_{k,f}^{(j-1)} (\mathbf{S}_{k,f}^{(j-1)})^T)^{-1} (\Delta\hat{\chi}_{k|k,f}^j) + (\Delta\mathbf{z}_{k,f}^j)^T \mathbf{R}_c^{-1} \Delta\mathbf{z}_{k,f}^j \quad (5.30)$$

$$J(\hat{\chi}_{k|k,f}^j) = (\Delta\hat{\chi}_{k|k,f}^j)^T \mathbf{Y}_{k,f}^{(j-1)} \Delta\hat{\chi}_{k|k,f}^j + (\Delta\mathbf{z}_{k,f}^j)^T \mathbf{R}_c^{-1} \Delta\mathbf{z}_{k,f}^j \quad (5.31)$$

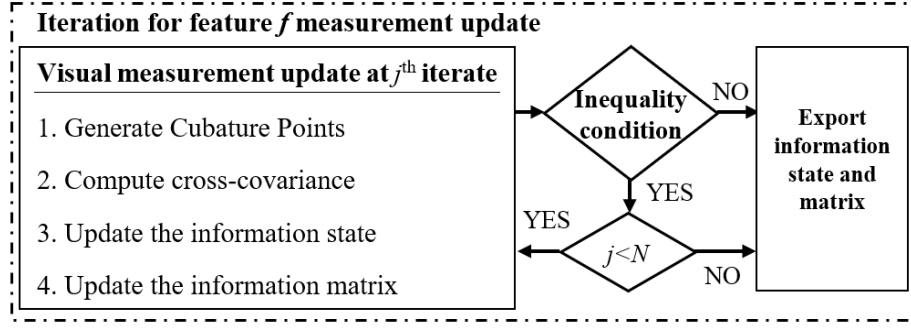


Figure 5.4: The iteration process for feature f measurement update on one core

5.6 Experimental Validation

This section introduces a method to implement the VIO parallel processing easily with popular software. For this purpose, the implementation utilizes the `parallel computing toolbox` in MATLAB, which provides many advantages to convert the original serial-computing code to the parallel-computing code. Algorithm 1 presents pseudo code to implement parallel processing. The usual `for`-loop is replaced by the `parfor`-loop, where

each loop iteration will process one feature in nondeterministic order. Hence, the original code needs to be modified such that each `parfor`-loop iteration is independent. The library also allows the implementation of a `while`-loop inside the `parfor`-loop. The `while`-loop is the place where the optimization procedure is implemented for each feature measurement update. The synthesis of all individual feature evaluations ($\mathbf{Y}_{k,f}$ and $\mathbf{y}_{k,f}$) is performed using (5.29). The proposed algorithm is evaluated in terms of the estimation error and the processing time in the experiments with some publicly available datasets.

Algorithm 2: Pseudo code for parallel processing

```

1 Function parfor ( $\mathbf{Y}_{k|k-1}$ ,  $\mathbf{y}_{k,k-1}$ ,  $\mathbf{z}_{k,f}$ ,  $\hat{\chi}_{k|k,f}^j$ ,  $\mathbf{R}_c$ ) :
2   while (inequality condition is satisfied) do
3     Compute  $\hat{\mathbf{z}}_{k,f}^j$  and evaluate  $\mathbf{Y}_{k,f}$ ,  $\mathbf{y}_{k,f}$ 
4     Evaluate the cost function  $J(\hat{\chi}_{k|k,f}^j)$ 
5     Check inequality condition
6   end
7   return  $\mathbf{Y}_{k,f}$ ,  $\mathbf{y}_{k,f}$ 

```

The proposed algorithm is implemented on different hardware computing platforms: a generic laptop, a mini PC kit and a powerful desktop. The technical specifications of these hardwares are described in Table 5.1. Thanks to the hyper-threading application, two logical processing units (a.k.a. cores) are deployed on one physical core when implementing parallel processing. The maximum number of logical cores is 8 (laptop), 8 (Intel®NUC) and 24 (desktop). The experiment helps determine whether the deployment of parallel processing is beneficial to the VIO system.

Table 5.1: Technical specifications of hardware platforms

| | Laptop | Mini PC kit | Desktop |
|------------------------------------|-------------------------------|------------------------------|----------------------------|
| Device Model | Inspiron 15 7000 | Intel®NUC NUC7i7BNH | Precision T5500 |
| CPU model | Intel®Core™ i7-7700HQ [14] | Intel®Core™ i7-7567U [13] | Intel® Xeon® x5680 [12] |
| # of Physical Cores | 4 | 4 | 12 |
| # of Logical Cores | 8 | 8 | 24 |
| CPU Cache (MB) | 6 | 4 | 12 |
| Processor Based Frequency (GHz) | 2.8 | 3.5 | 3.33 |
| Bus Speed (GT/s) | 8 | 4 | 6.4 |
| RAM (GB) | 16 | 16 | 48 |

For our purpose, the KITTI dataset [56] is utilized to verify the proposed algorithm. On average, 40024 bytes are sent to one core for processing, and 85632 bytes are received. The parallel implementation can be evaluated using two criteria: speed-up (5.32) and efficiency (5.33) [111].

$$Speedup = \frac{T_{serial}}{T_{parallel}} \quad (5.32)$$

$$Efficiency = \frac{Speedup}{N_{core}} \quad (5.33)$$

where T_{serial} is the processing time of serial computation; $T_{parallel}$ is the processing of parallel computation; N_{core} is the number of cores used in parallel computing. The evaluation helps to determine whether the use of more cores is efficient to accelerate the computation.

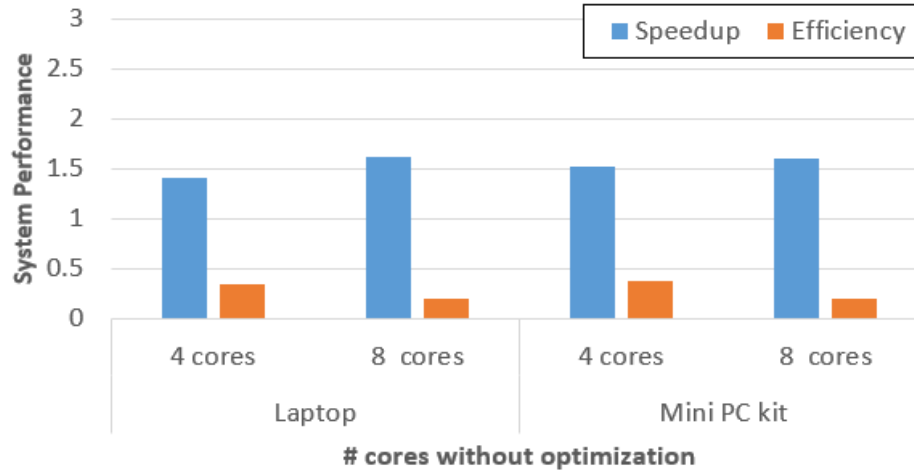


Figure 5.5: Speed-up and efficiency evaluation on laptop and mini PC kit without optimization process

In Figs.5.5, 5.6 and 5.7, the use of more cores has accelerated the computation. For example, the processing time of the laptop implementation has decreased by 29.0% with 4 cores and by 38.1% with 8 cores, compared to serial processing. That speed-up is smaller

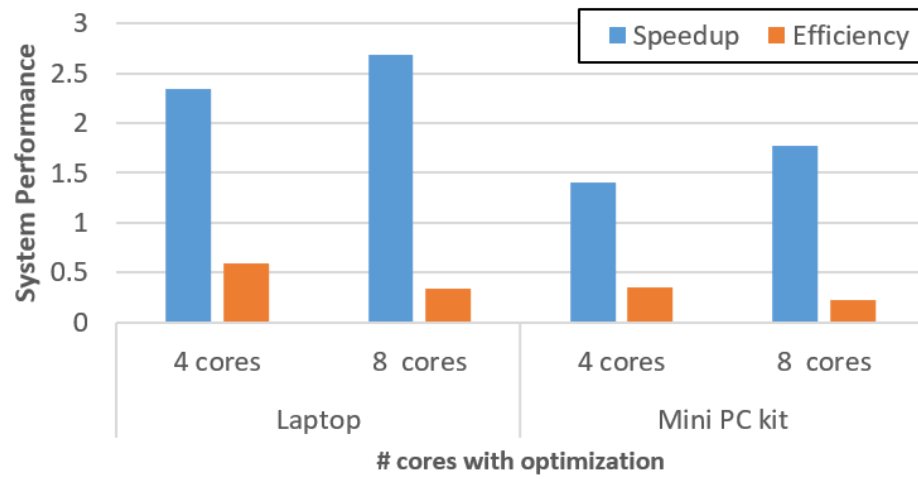


Figure 5.6: Speed-up and efficiency evaluation on laptop and mini PC kit with optimization process

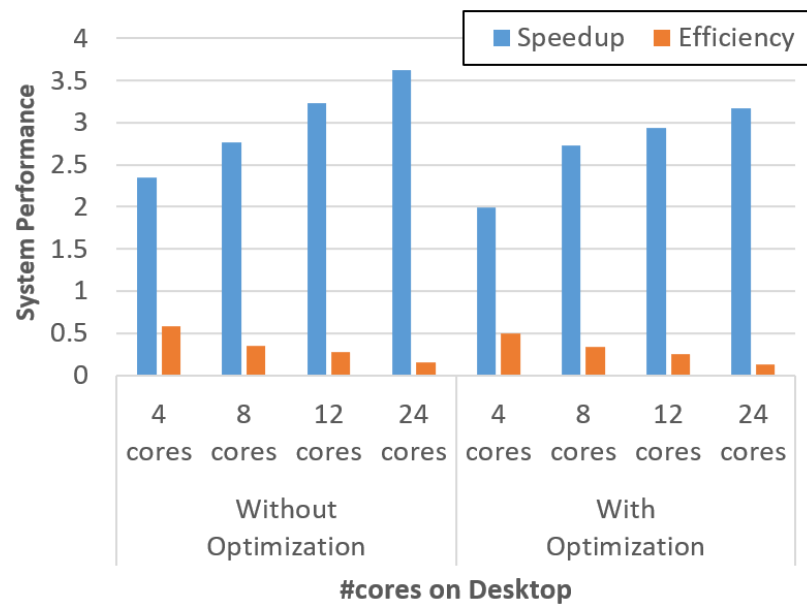


Figure 5.7: Speed-up and efficiency evaluation on desktop

than the ideal speed-up of a factor of 4 on 4 cores, and a factor of 8 on 8 cores. Similar results are observed in the experiments with Intel[®]NUC and desktop. In the desktop evaluation (Fig.5.7), even though the implementation deploys 24 logical cores for execution, the processing time is not reduced significantly compared to that of 8 cores. This phenomenon happens because of parallel overhead, which includes the time for transferring data to each core and back using limited cache memories. The parallel overhead results in a decrease of acceleration efficiency when employing more cores for computation.

According to the theoretical structure of the information filter [32, 57], the prediction step uses covariance (**P**) while the update step employs information matrix (**Y**). The system architecture (Fig.5.3) requires the unavoidable transferring between the covariance matrix and information matrix, which may increase the processing time. However, the experimental results with the computational acceleration have shown that the transfer does not dismiss the parallelization benefits. The deployment of more cores also results in an increase in power consumption. For example, the laptop consumes 3.8W (serial), 42.1W (4 cores) and 36.6W (8 cores). Compared to serial processing, the power consumption has increased $2.1\times$ with 4 cores and $2.6\times$ with 8 cores. Hence, the deployment of more cores may negatively affect the power consumption of the vehicle. In addition, the implementation of the information filter requires the transfer between the state and information form, as in Fig.5.3. Despite the transfer, the parallel computing on the information domain is able to accelerate the VIO computation.

The KITTI dataset allows verifying the system performance in a long-term operation with long-distance traveling. It also helps to observe the impact of the accumulative errors as well as the effect of implementing the optimization for each feature measurement update. In the KITTI dataset 2011_09_30_0034, the vehicle travels about 900m at 25km/h.

Root mean square error and rotation error are used to evaluate the estimation accuracy performance. The iterated CIF keeps track of the vehicle trajectory (Fig.5.8) and continuously updates the IMU bias (Fig.5.12) and the vehicle velocity (Fig.5.11). In this case, the iterated CIF has a superior performance than does the CKF filter. The employment of the optimization process has effectively eliminated the accumulative errors and decreased the estimation errors. In Fig.5.10, the rotation estimates of these filters have similar accuracy.

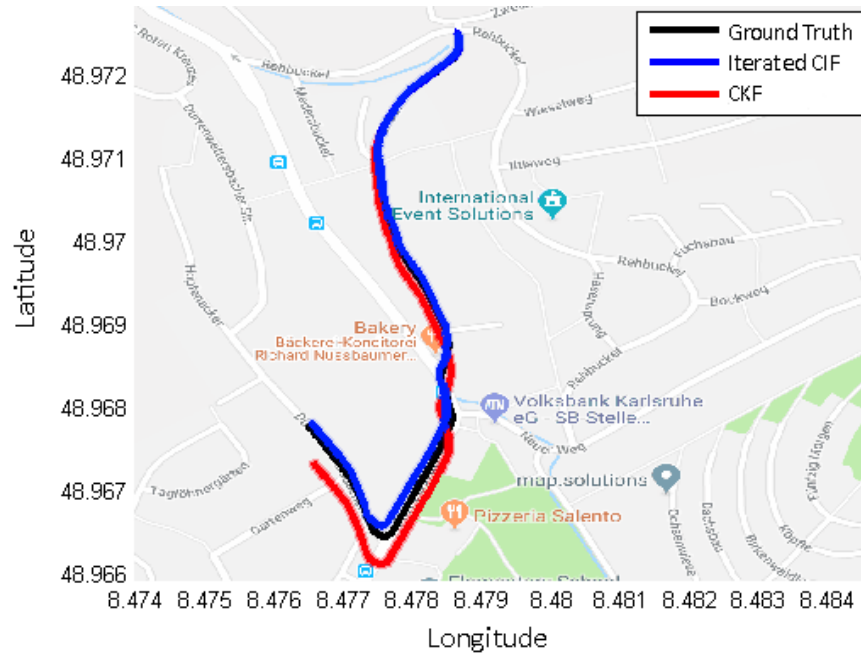


Figure 5.8: Position estimates of the KITTI dataset 2011_09_30_0034

Figures 5.13, 5.14 and 5.15 present the experiment with dataset 2011_09_30_0033. The vehicle travels in a loop about $\sim 1800\text{m}$ at $\sim 45\text{km/h}$. Considering the RMSE evaluation (Fig.5.14), both filters show almost comparative estimations in this case. The employment of optimization does not help to improve the estimation effectively. The optimization step

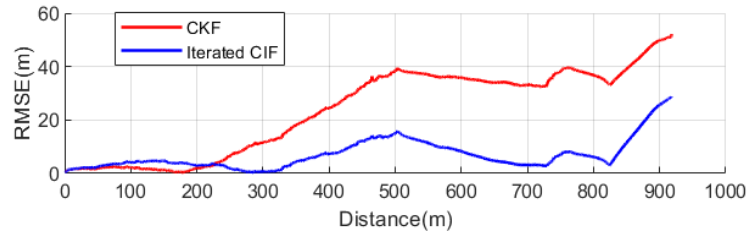


Figure 5.9: RMSE evaluation for dataset 2011_09_30_0034

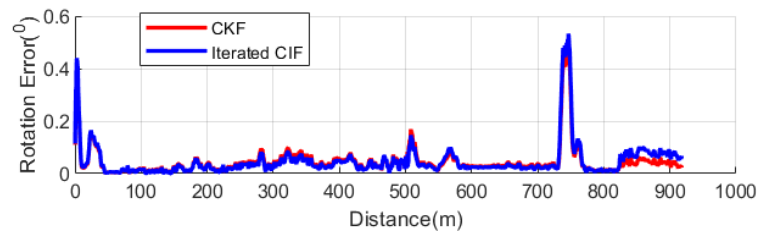


Figure 5.10: Rotation error evaluation for dataset 2011_09_30_0034

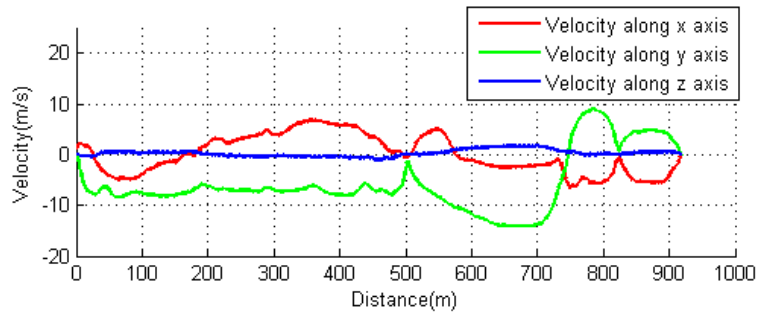


Figure 5.11: Iterated CIF estimation of velocity with dataset 2011_09_30_0034

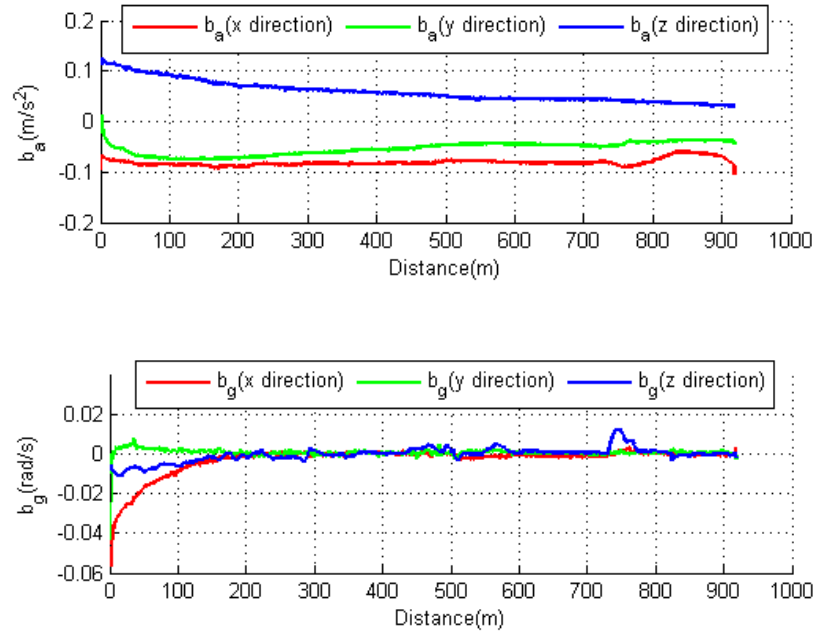


Figure 5.12: Iterated CIF estimation of the accelerometer bias and gyroscope bias with 2011_09_30_0034

with MLE cannot bring the estimate up to the likelihood surface due to the measurement noise. The MLE based termination criteria are not powerful enough to cover most of the situations. Considering the rotation accuracy in Fig.5.15, two filters have similar accuracy.

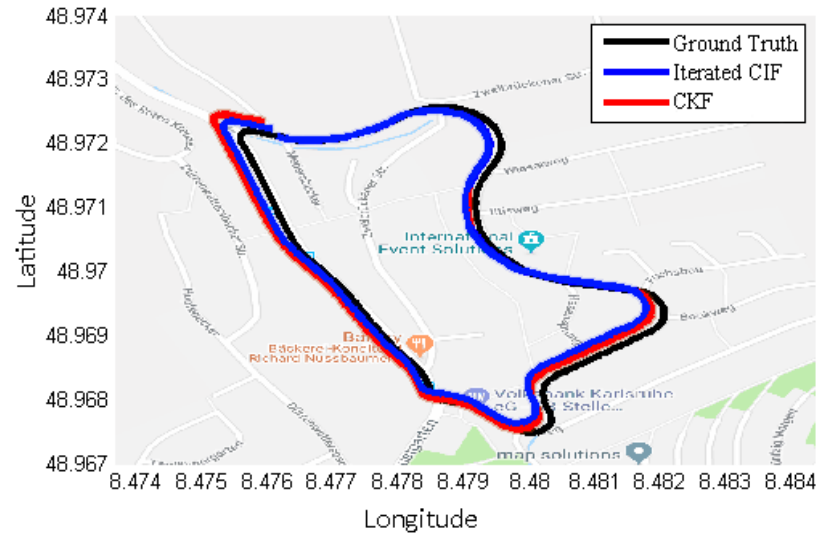


Figure 5.13: Position estimates of the KITTI dataset 2011_09_30_0033

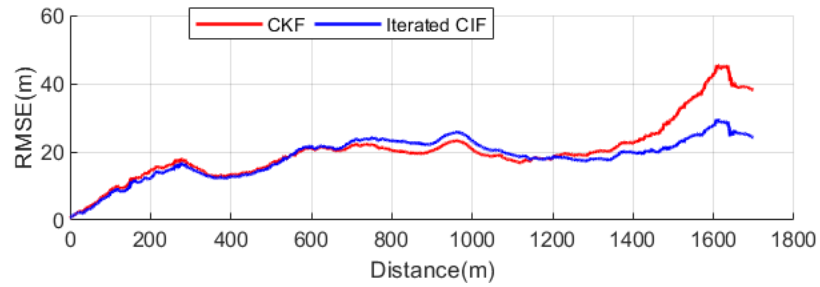


Figure 5.14: RMSE evaluation for dataset 2011_09_30_0033

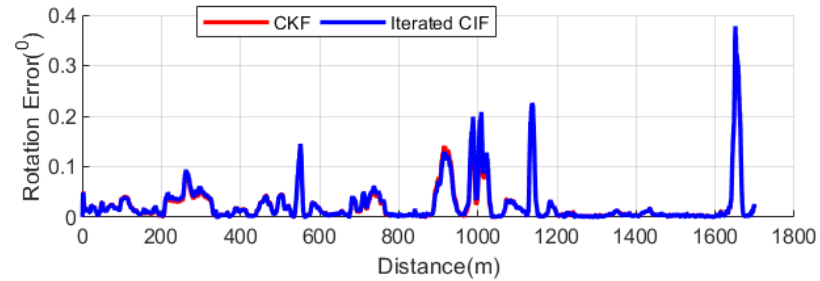


Figure 5.15: Rotation error evaluation for dataset 2011_09_30_0033

The proposed algorithm is also verified with the EuRoC dataset [26] (Fig.5.16), which is collected in an indoor environment using an unmanned aerial vehicle. The EuRoC dataset only allows the verification within a short distance (about 100m per sequence). In Figures 5.17 and 5.18, the iterated CIF estimation is compared with VINS-Mono [119], which is a global-optimization based approach with a loop closure function. In the RMSE evaluation of positional estimation, the iterated CIF accuracy is comparable to the VINS-Mono in the experiment. VINS-Mono performs full smoothing, which estimates for the entire history of camera poses and 3D landmarks. The proposed algorithm is classified as a filtering approach, which only focuses on inferring the current state directly from the sensor data. Compared to VINS-Mono, our computation does not attempt to solve the optimization problem and generate the map of 3D points at each time instant, which requires fewer computational resources and processing times. The implementation does not require the installation of an advanced optimization library such as Google’s Ceres solver [16]. The proposed algorithm is beneficial to micro robotic systems having limited computational capability.

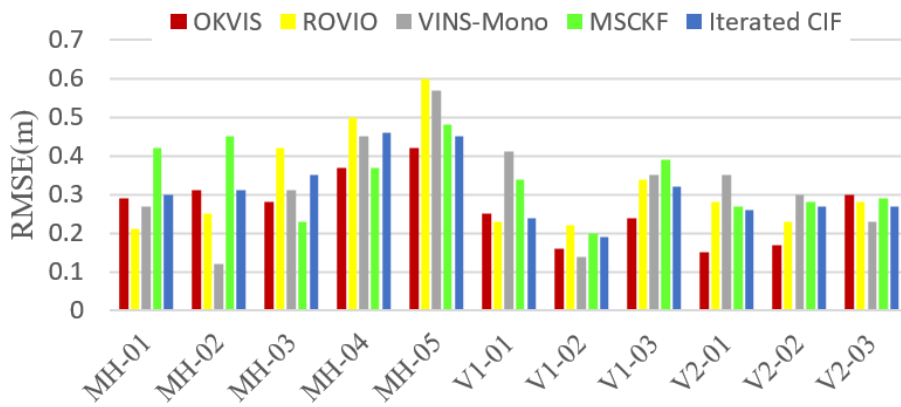


Figure 5.16: RMSE evaluation for 11 sequences of EuRoC dataset [26]

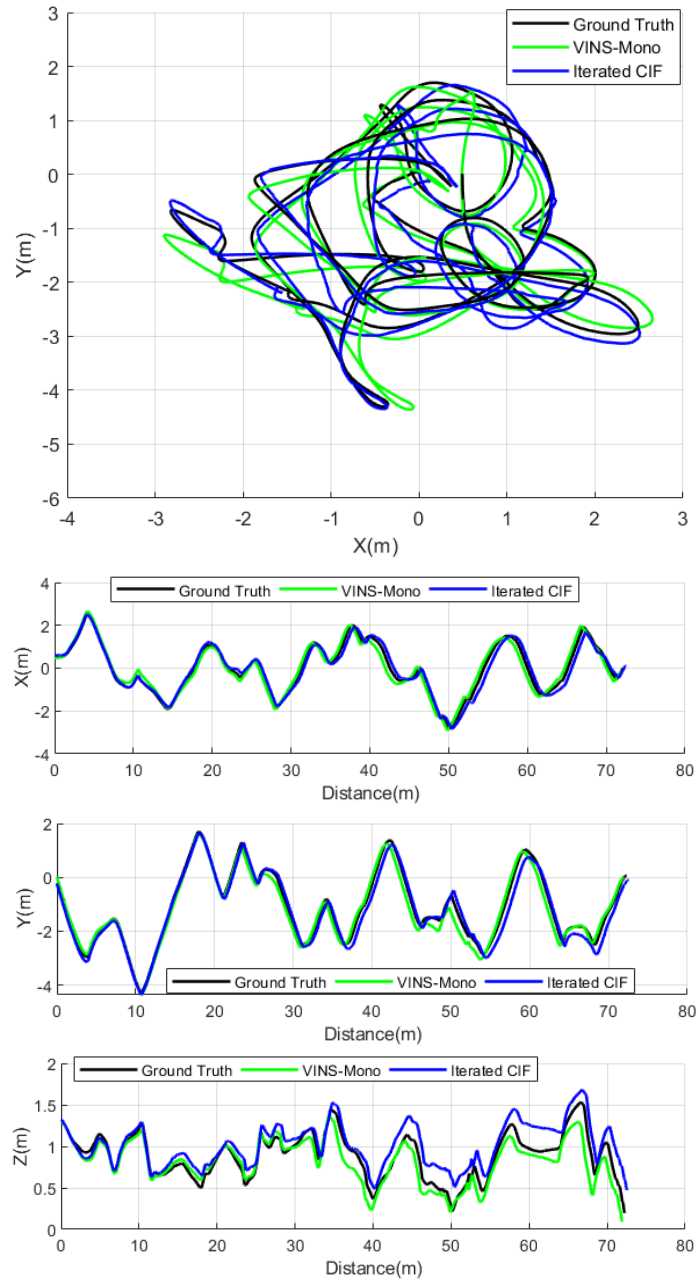


Figure 5.17: Experimental results of EuRoC dataset V1_02_medium with RMSE evaluation of positional estimation

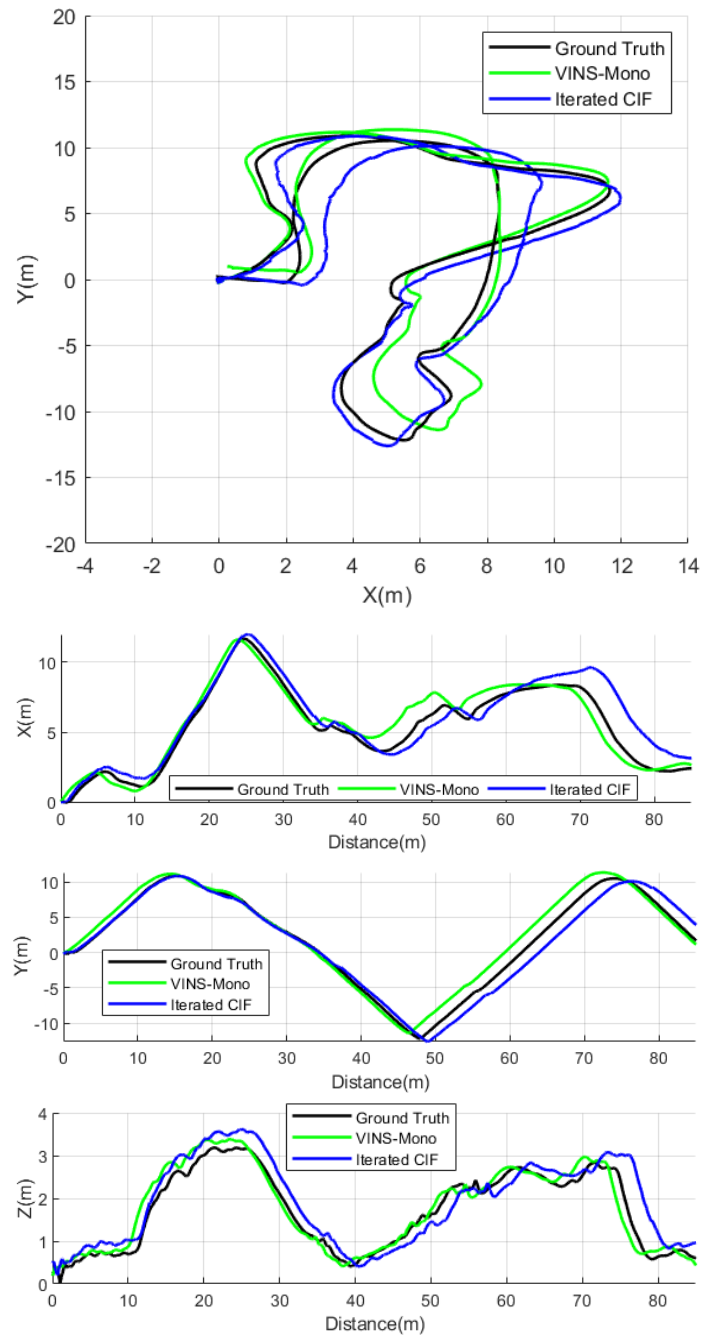


Figure 5.18: Experimental results of EuRoC dataset MH.04_difficult with RMSE evaluation of positional estimation

The proposed algorithm can be implemented in specialized hardware platforms such as GPU. The structure of GPU offers more degrees of parallelism with hundreds of cores. However, the GPU communicates with the host CPU via the PCI Express bus, which limits the data transfer rate. Consequently, the use of more core in GPU cannot achieve an expected efficiency of speeding up despite the utilization of many cores. Programming for the GPU is also challenging to achieve peak execution performance.

5.7 Conclusions

This chapter presented the parallel processing implementation for visual inertial odometry with a multiple-core CPU. The visual measurement update utilizes the CIF to distribute the computation to multiple cores and evaluates the contribution of each feature to the motion's estimation. When processing each feature independently, the iteration process is applied to perform MLE based optimization. This helps to minimize the error accumulation and improve the estimation accuracy over long-term operation. The deployment of more cores significantly reduces the processing time of the expensive optimization process.

In order to evaluate the efficiency of the parallel processing, the proposed algorithm is implemented on different hardware computing platforms with 4 to 24 cores. The limitation of data transfer on the multi-core CPU structure prevented the system from achieving the ideal speed-up. Additionally, the system also requires more power consumption, which limits its usefulness for some micro- or nano- robot applications.

To handle the system nonlinearity, the CIF has generated a minimal set of sampling points, which are propagated through the nonlinear functions. This strategy results in a better approximation of the mean and covariance. The filter design follows the invariant

Kalman filtering, which in turn achieves the system's consistency. The algorithm estimation is comparable to other available algorithms.

Overall, the expensive visual measurement update of VIO is distributed on multiple cores, which in turn accelerates the computation. The acceleration was verified on a generic computer with a popular MATLAB parallel computing toolbox. This easy setup suggests a method to test the VIO parallel processing before deploying it on specialized hardware computing platforms.

Chapter 6

Conclusion and Outlook

This chapter summarizes the main contributions of the research described in this thesis and a number of possible extensions to the research are discussed.

The primary concentration of this research was to address the sensor fusion problem associated with visual inertial odometry in GPS-denied environments. This research has led to multiple implementations of a cubature Kalman filter-based solution to reduce the system complexity, improve computational efficiency and guarantee an acceptable error of estimation. The research was based on three primary objectives:

1. To design a nonlinear filtering algorithm for VIO sensor fusion using a straightforward TTG-based visual measurement model,
2. To address the error accumulation of the VIO estimation over long-term operation,
3. To enhance parallel processing for VIO computation with multi-core CPUs.

The research contributions related to each objective are summarized in the following sections.

6.1 Research Summary Based on Objective I

For the first objective, a VIO sensor fusion algorithm to integrate IMU data and camera information was developed. After an extensive literature review, it was concluded that the number of features, the number of camera poses, and the procedure to predict the visual measurement dramatically affect the system complexity and computation cost of execution [68, 125]. At each time instant, for one landmark feature, the system needs to perform a least square Gauss-Newton optimization to estimate the 3D feature point before predicting the visual measurement. To reduce the computational cost in many cases, the VIO process selected a small subset of features. The point transfer approach using TTG was able to predict the visual measurement using only three camera frames. To this end, a visual measurement model based on TTG was developed, which was a non-recursive highly-nonlinear function. In the experimental comparison, the TTG-based measurement model and the traditional model produced similar predictions of the visual measurements when provided the same inputs. For the development of the VIO sensor fusion system, the CKF was formulated for VIO application to handle the system nonlinearity. The proposed algorithm was validated by a set of publicly-available datasets and compared with other existing VIO algorithms. Additionally, CIF for VIO applications was developed. CIF implementation avoided the inverse computation of the high-dimensional innovation covariance matrix. This step helped to further improve computational efficiency. Overall this algorithm was implemented as a fast VIO solution for high-speed autonomous robotic systems.

6.2 Research Summary Based on Objective II

Secondly, the error accumulation in the VIO estimation due to noise in camera measurement was addressed. This made the estimation error diverge over long-term operations. As shown in the literature, global pose graph optimization and loop closure were the most common techniques to minimize the accumulative errors. However, these techniques demanded relatively high computational resources and processing time, which was contrary to objective 1 in terms of computational efficiency. To this end, instead of optimizing for the entire trajectory, optimization was performed for the latest filter state and covariance during the measurement update. The success of the solution was verified by practical experiments; it allowed the filter to operate for a longer duration. This solution was suitable for the self-localization project and did not require any map or the installation of advanced optimization library software. For larger navigation projects, the second solution was developed to integrate the VIO estimation with pseudo-ranging measurement. The experimental results showed that the integration helped to bound the estimation errors over long-term operations.

6.3 Research Summary Based on Objective III

The third objective of this research was to speed up the VIO execution through hardware technique. In the literature, many advanced hardware computing platforms have been employed to perform parallel processing [157]. However, the availability of these specialized hardware is limited and most researchers utilize generic multi-core computers to develop their algorithms. Also, the multiple-core computer is becoming increasingly popular in

robotic systems. A multi-core CPU was used for parallel processing. Individual features were processed independently, which in turn reduced the processing time significantly. The filtering structure also utilized an invariant Kalman filter, which achieved consistent performance. The proposed algorithm was validated using various multi-core hardware platforms, such as the general laptop, mini desktop (Intel®NUC) and powerful-computing desktop. Moreover, the iteration procedure was applied to minimize the error accumulation of the VIO estimate over long-term operation. This application did not increase the processing time significantly, due to the deployment of parallel processing, while guaranteeing similar estimation errors and stability.

6.4 Summary of Contributions

To summarize, this thesis describes research that made the following key contributions in visual inertial odometry, fulfilling all of the outlined research objectives.

1. Contributions related to Objective I:

- A design of a straightforward visual measurement model using trifocal tensor geometry.
- A comparison between the traditional measurement model and the TTG-based measurement model.
- A novel formulation of a cubature Kalman filter for VIO application.
- A novel formulation of a cubature information filter for VIO application.

2. Contributions related to Objective II:

- A solution to address the VIO error accumulation for a self-localization project using an iterated cubature Kalman filter.
- A solution to address the VIO error accumulation for larger localization projects using ranging measurements.

3. Contributions related to Objective III:

- A novel approach to accelerate VIO execution using multi-core CPU.
- An application of an invariant cubature information filter for VIO applications.
- A solution to perform local optimization within a VIO filtering system to address the error accumulation over long-term operation.
- Validation of the proposed algorithm on different hardware multi-core computing platforms (4 to 24 cores) to evaluate the efficiency of the acceleration.

This work described in this thesis was awarded:

- Wally Read Best Young Professionals Paper in Newfoundland Electrical and Computer Engineering Conference, St.John's, Newfoundland, November 15, 2017.
- First prize of poster presentation in Annual Research Day, Faculty of Engineering & Applied Science, Memorial University, 2017.
- Second prize of video presentation in Annual Research Video Communications Award, Faculty of Engineering & Applied Science, Memorial University, 2016.

This research led to the following scientific articles:

1. T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "Developing Computationally-Efficient Nonlinear Cubature Kalman Filtering for Visual Inertial Odometry", ASME Journal of Dynamic Systems, Measurement and Control, February 2019.
2. T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "Accelerating Visual Inertial Odometry Using Parallel Processing and Cubature Information Filter", IEEE/ASME Transactions on Mechatronics, (Under Review), 2019.
3. T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "CKF-Based Visual Inertial Odometry for Long-Term Trajectory Operations", ASME Journal of Dynamic Systems, Measurement and Control (Under Review), 2019.
4. T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "Likelihood-based Iterated Cubature Multi-State-Constraint Kalman Filter for Visual Inertial Navigation System", IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), October 2017.
5. T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "Developing Moving Horizon Estimation Based Ranging Measurement for Supporting Vision-Aided Inertial Navigation System", ASME Dynamic Systems and Control Conference (DSCC), November 2017.
6. T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "Developing a Cubature Multi-State Constraint Kalman Filter for Visual-Inertial Navigation System", Canadian Conference on Computer and Robot Vision (CRV), May 2017.
7. T. Nguyen, G. K. I. Mann, A. Vardy, R. G. Gosine, "Computationally-Efficient Visual Inertial Odometry for Autonomous Vehicles", IEEE Newfoundland Electrical and

6.5 Future Research Directions

The research presented in this thesis has a number of potential extensions. These future developments aim at improving its practicality in real-world applications.

Visual inertial navigation applied for hardware-constrained systems: Micro robotic systems have limitations of CPU, memory and power resources. The implementation of the computationally-efficient VIO algorithm allows the system to self-localize in GPS-denied environment and operate in longer duration. Also, the proposed VIO can be applied for some virtual reality and augmented-reality applications with mobile and wearable devices such as smart phone and Google Glass. The implementation will address the issue of robust recovery, observability constraints and initialization schemes for particular devices. Depending on the specific application, future research also investigates the case when the system is unable to perform visual measurement update temporarily, such as a hovering state of a micro aerial vehicle.

Integration with other advanced sensors: The proposed VIO can be employed in larger navigation projects, where there are supports from other advanced sensors. Combination with these sensors, such as LIDAR and laser range, can help to improve the estimation accuracy and robust performance in a dynamic environment. The effectiveness of the combination depends on the strategy to align multiple estimations [29, 144]. A future research direction could focus on developing the synthesis algorithm. It can be constructed as an optimization problem or in the scheme of Kalman filtering for multi-sensor systems. In the first option, all sensors have a similar role in the system with different constraints.

The optimizer may require considerable computational resources for execution. The failure of one sensor may greatly affect to the estimation results. The Kalman filter based structure provides the greater flexibility in the system operation. The VIO will play the primary role of the navigation system and receive support from other sensors.

Improving the visual measurement update: Experimental results have illustrated the benefit of using local optimization to improve estimation accuracy over long-term operation. The optimization can be performed in the absence of a map. As mentioned in chapter 4, the effectiveness of the optimization depends on the criteria to terminate the iteration procedure. Future research can be undertaken to develop better strategies for local optimization within the VIO system. On the other hand, the use of more than 3 camera images is another possibility to improve filter performance. This modification results in more reliable feature points for the filter update. Additionally, more TTG based geometric constraints can be utilized when designing the visual measurement model.

Parallel processing for each feature point: Future studies can expand the parallel-processing approach to process each feature more effectively and select features which contribute most to the trajectory estimation. In a dynamic environment, the contribution of each feature to the filter estimation is not identical due to different observation uncertainty parameters [112, 114]. These parameters were assumed to be identical in the filter implementation. This assumption can be removed using the structure of parallel processing. Each feature would be processed independently with different measurement noise parameters, which can be defined using an observation-covariance estimator or machine learning. Additional computation would lead to the requirement for a training phase and an increase of computational cost. This issue should also be considered in future research.

Bibliography

- [1] Casadi library. <https://github.com/casadi/casadi/wiki>.
- [2] Cyberhawk. <https://www.thecyberhawk.com/>.
- [3] Decawave dwm1000 module. <http://www.decawave.com/products/dwm1000-module>.
- [4] Intel euclid developer kit. <https://click.intel.com/intelr-euclidtm-development-kit.html>.
- [5] Intel zr300 developer kit. <https://click.intel.com/intelr-realsensetm-development-kit-featuring-the-zr300.html>.
- [6] Matlab code examples of CKF. <https://haranarasaratnam.com/software.html>.
- [7] Microdrones. <https://www.microdrones.com/en/industry-experts/>.
- [8] Odroid platforms. http://www.hardkernel.com/main/products/prdt_info.php.
- [9] Optor visual inertial camera. <https://www.robotshop.com/ca/en/optor-visual-inertial-camera.html>.

- [10] Parrot s.l.a.m.dunk. <https://www.parrot.com/global/business-solutions/parrot-slamdunk>.
- [11] The sky guys. <https://theskyguys.ca/contact/>.
- [12] Technical Specifications of a Intel Xeon Processor X5680. <https://ark.intel.com/products/47916/Intel-Xeon-Processor-X5680-12M-Cache-3-33-GHz-6-40-GT-s-Intel-QPI->.
- [13] Technical Specifications of Intel Core i7-7567U Processor. <https://ark.intel.com/products/97541/Intel-Core-i7-7567U-Processor-4M-Cache-up-to-4-00-GHz->.
- [14] Technical Specifications of Intel Core i7-7700HQ Processor. <https://ark.intel.com/products/97185/Intel-Core-i7-7700HQ-Processor-6M-Cache-up-to-3-80-GHz->.
- [15] Time domain's p-410 uwb module. <http://www.timedomain.com/p400-mrm.php>.
- [16] S. Agarwal, K. Mierle, and Others. Ceres solver. <http://ceres-solver.org>.
- [17] N. Andreff and B. Tamadazte. Laser steering using virtual trifocal visual servoing. *International Journal of Robotics Research*, 35(6), 2015.
- [18] I. Arasaratnam and S. Haykin. Cubature kalman filters. *IEEE Transactions on Automatic Control*, 54(6), 2009.
- [19] A. Barrau and S. Bonnabel. The Invariant Extended Kalman Filter as a Stable Observer. *IEEE Transactions on Automatic Control*, 62(4), 2017.

- [20] A. Barrau and S. Bonnabel. Invariant Kalman Filtering. *Annual Reviews of Control, Robotics, and Autonomous Systems*, (December 2017):1–21, 2018.
- [21] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, (3):346–359, June 2008.
- [22] D. S. Bayard and P. B. Brugarolas. An Estimation Algorithm for Vision-Based Exploration of Small Bodies in Space. In *Proc. of American Control Conference*, 2005.
- [23] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart. Iterated extended kalman filter based visual-inertial odometry using direct photometric feedback. *The International Journal of Robotics Research*, 36(10):1053–1072, 2017.
- [24] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart. Robust Visual Inertial Odometry Using a Direct EKF-Based Approach. In *In Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.
- [25] M. Bloesch, S. Omari, H. Sommer, C. Gehring, J. Hwangbo, M. A. Hoepflinger, M. Hutter, and R. Siegwart. Fusion of Optical Flow and Inertial Measurements for Robust Egomotion Estimation. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014.
- [26] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart. The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 2016.
- [27] F. Camposeco and M. Pollefeys. Using vanishing points to improve visual-inertial odometry. In *Proc. of IEEE International Conference on Robotics and Automation*, 2015.
- [28] L. Carlone and S. Karaman. Attention and Anticipation in Fast Visual-Inertial Navigation. In *Proc. of IEEE International Conference on Robotics and Automation*, 2017.

- [29] D. Caruso, A. Eudes, M. Sanfourche, and D. Vissi. Robust indoor / outdoor navigation through magneto-visual-inertial optimization-based estimation. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.
- [30] E. Celledoni, H. Marthinsen, and B. Owren. An introduction to Lie group integrators basics, new developments and applications. *Journal of Computational Physics*, 257(November), 2014.
- [31] J. Ćesić, I. Marković, M. Bukal, and I. Petrović. Extended information filter on matrix Lie groups. *Automatica*, 82(April), 2017.
- [32] K. P. B. Chandra, D.-W. Gu, and I. Postlethwaite. Cubature information filter and its applications. *Proc. of American Control Conference*, 2011.
- [33] K. P. B. Chandra, D. W. Gu, and I. Postlethwaite. Cubature H_∞ information filter and its extensions. *European Journal of Control*, 2016.
- [34] Y. Chen, G. L. Yang, Y. X. Jiang, and X. Y. Liu. Monocular Visual Odometry Based on Trifocal Tensor Constraint Monocular. *Journal of Physics: Conference Series*, 2018.
- [35] J. Civera, O. G. Grasa, A. J. Davison, and J. M. M. Montiel. 1-Point RANSAC for EKF-Based Structure from Motion. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.
- [36] J. Civera, O. G. Grasa, A. J. Davison, and J. M. M. Montiel. 1point ransac for extended kalman filtering: Application to realtime structure from motion and visual odometry. *Journal of Field Robotics*, 2010.
- [37] B. Claus, J. H. Kepper IV, S. Suman, and J. C. Kinsey. Closed-loop one-way-travel-time navigation using low-grade odometry for autonomous underwater vehicles. *Journal of Field Robotics*, (July), 2017.

- [38] L. Clement, V. Peretroukhin, and J. Kelly. Improving the Accuracy of Stereo Visual Odometry Using Visual Illumination Estimation. In *Proc. of International Symposium on Experimental Robotics*, 2016.
- [39] L. E. Clement, V. Peretroukhin, J. Lambert, and J. Kelly. The battle for filter supremacy a comparative study of the multi-state constraint kalman filter and the sliding window filter. In *Proc. of 12th Conference on Computer and Robot Vision*, 2015.
- [40] D. Crisan and O. O. Particle filters with random resampling times. *Stochastic Processes and Their Applications*, 122(4), 2012.
- [41] N. De Palézieux, T. Nägeli, and O. Hilliges. Duo-VIO: Fast, light-weight, stereo Inertial Odometry. In *Proc. of IEEE International Conference on Intelligent Robots and Systems*, 2016.
- [42] O. De Silva, G. K. I. Mann, and R. G. Gosine. An Ultrasonic and Vision-based Relative Positioning Sensor for Multi-Robot Localization. *IEEE Sensors Journal*, 2014.
- [43] H. Deilamsalehy, T. C. Havens, and J. Manela. Heterogeneous Multisensor Fusion for Mobile Platform Three-Dimensional Pose Estimation. *ASME Journal of Dynamic Systems, Measurement, and Control*, 2017.
- [44] J. Delmerico and D. Scaramuzza. A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots. In *Proc. of IEEE International Conference on Robotics and Automation*, 2018.
- [45] D. D. Diel. *Stochastic Constraints for Vision-Aided Inertial Navigation*. PhD thesis, Massachusetts Institute of Technology, 2005.

- [46] T.-C. Dong-Si and A. I. Mourikis. Motion tracking with fixed-lag smoothing: Algorithm and consistency analysis. In *Proc. of IEEE International Conference on Robotics and Automation*, 2011.
- [47] H. Durrant-Whyte. Multi sensor data fusion. Course Notes, Australian Centre for Field Robotics, The University of Sydney, 2001.
- [48] J. Engel, T. Schops, and D. Cremers. LSD-SLAM: Large-Scale Direct Monocular SLAM. In *Proc. of European Conference on Computer Vision*, 2014.
- [49] J. Engel, J. Sturm, and D. Cremers. Camera-based navigation of a low-cost quadrocopter. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2815–2821, Oct. 2012.
- [50] W. Fang, L. Zheng, H. Deng, and H. Zhang. Real-Time Motion Tracking for Mobile Augmented/Virtual Reality Using Adaptive Visual-Inertial Fusion. *Sensors*, 17, 2017.
- [51] M. A. Fischler and R. C. Bolles. Random sample consensus, a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 1981.
- [52] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. Imu preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation. In *Proc. of Robotics: Science and Systems (RSS)*, 2015.
- [53] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. On-Manifold Preintegration for Real-Time Visual Inertial Odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2017.
- [54] C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *Proc. of IEEE International Conference on Robotics and Automation*, 2014.

- [55] R. Garcia, J. Puig, P. Ridao, and X. Cufi. Augmented State Kalman Filtering for AUV Navigation. In *Proc. of IEEE International Conference on Robotics & Automation*, number May, 2002.
- [56] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.
- [57] B. Grocholsky, A. Makarenko, and H. Durrant-Whyte. Information-theoretic coordinated control of multiple sensor platforms. In *Proc. of IEEE International Conference on Robotics and Automation*, 2003.
- [58] C. Guo, D. Kottas, R. DuToit, A. Ahmed, R. Li, and S. Roumeliotis. Efficient Visual-Inertial Navigation using a Rolling-Shutter Camera with Inaccurate Timestamps. *Robotics: Science and Systems*, 2014.
- [59] K. E. Hamilton, C. D. Schuman, S. R. Young, N. Imam, and T. S. Humble. Neural networks and graph algorithms with next-generation processors. In *Proc. of IEEE International Parallel and Distributed Processing Symposium Workshops*, 2018.
- [60] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge, 2004.
- [61] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis. Camera imu based localization: Observability analysis and consistency improvement. *The International Journal of Robotics Research*, 33(1), 2014.
- [62] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis. Consistency analysis and improvement of vision aided inertial navigation. *IEEE Transactions on Robotics*, 30(1), 2014.

- [63] T. Hinzmann, T. Schneider, M. Dymczyk, A. Schaffner, S. Lynen, R. Siegwart, and I. Gilitschenski. Monocular Visual-Inertial SLAM for Fixed-Wing UAVs Using Sliding Window Based Nonlinear Optimization. In *Proc. of International Symposium on Visual Computing*, 2013.
- [64] J.-S. Hu and M.-Y. Chen. A sliding-window visual-imu odometer based on tri-focal tensor geometry. In *Proc. of IEEE International Conference on Robotics and Automation*, 2014.
- [65] V. Indelman, S. Williams, M. Kaess, and F. Dellaert. Information Fusion in Navigation Systems via Factor Graph Based Incremental Smoothing. *Journal of Robotics and Autonomous Systems*, 2013.
- [66] T. A. Johansen, J. M. Hansen, and T. I. Fossen. Nonlinear Observer for Tightly Integrated Inertial Navigation Aided by Pseudo-Range Measurements. *ASME Journal of Dynamic Systems, Measurement, and Control*, 2018.
- [67] S.-h. Jung and C. J. Taylor. Camera Trajectory Estimation using Inertial Sensor Measurements and Structure from Motion Results. In *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2001.
- [68] C. Kanellakis and G. Nikolakopoulos. Survey on Computer Vision for UAVs: Current Developments and Trends. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 2017.
- [69] G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *Proc. of IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2007.
- [70] X. Kong, W. Wu, L. Zhang, and Y. Wang. Tightly-coupled stereo visual-inertial navigation using point and line features. *Sensors (Switzerland)*, 2015.

- [71] D. G. Kottas and S. I. Roumeliotis. An Iterative Kalman Smoother for Robust 3D Localization on Mobile and Wearable devices. In *Proc. of IEEE International Conference on Robotics and Automation*, 2015.
- [72] T. Laidlow, M. Bloesch, W. Li, and S. Leutenegger. Dense RGB-D-Inertial SLAM with Map Deformations. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.
- [73] A. Lambert, P. Furgale, T. D. Barfoot, and J. Enright. Field Testing of Visual Odometry Aided by a Sun Sensor and Inclinometer. *Journal of Field Robotics*, 29(3):426–444, 2012.
- [74] J. W. Langelaan. *State Estimation for Autonomous Flight in Cluttered Environments*. PhD thesis, Stanford University, 2006.
- [75] A. Ledergerber, M. Hamer, and R. D. Andrea. A Robot Self-Localization System using One-Way Ultra-Wideband Communication. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3131–3137, 2015.
- [76] D.-J. Lee. Nonlinear estimation and multiple sensor fusion using unscented information filtering. *IEEE Signal Processing Letters*, 15, 2008.
- [77] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart. Keyframe-Based Visual-Inertial SLAM Using Nonlinear Optimization. *The International Journal of Robotics Research*, 34(3), 2014.
- [78] S. Leutenegger, S. Lynen, M. Bosse, and P. Furgale. Keyframe-Based Visual-Inertial Odometry Using Nonlinear Optimization. *International Journal of Robotics Research*, 2015.
- [79] M. Li and A. I. Mourikis. High-precision, consistent EKF-based visual-inertial odometry. *International Journal of Robotics Research*, 32(6):690–711, 2013.

- [80] M. Li and A. I. Mourikis. Real-time motion tracking on a cellphone using inertial sensing and a rolling-shutter camera. In *Proc. of IEEE International Conference on Robotics and Automation*, 2013.
- [81] M. Li and A. I. Mourikis. Online temporal calibration for camera-IMU systems. *International Journal of Robotics Research*, 33:947–964, 2014.
- [82] Y. Lin, W. Wu, and T. Qin. Autonomous aerial navigation using monocular visual-inertial fusion. *Journal of Field Robotics*, 2018.
- [83] H. Liu, M. Chen, G. Zhang, H. Bao, and Y. Bao. ICE-BA : Incremental, Consistent and Efficient Bundle Adjustment for Visual-Inertial SLAM. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
- [84] H. Liu, R. Jiang, W. Hu, and S. Wang. Navigation drift analysis for visual odometry navigation. *Computing and Informatics*, 33, 2014.
- [85] T. Liu and S. Shen. High Altitude Monocular Visual-Inertial State Estimation : Initialization and Sensor Fusion. In *Proc. of IEEE International Conference on Robotics and Automation*, 2017.
- [86] T. Liu and S. Shen. Spline-Based Initialization of Monocular Visual Inertial State Estimators at High Altitude. *IEEE Robotics & Automation Letters*, 2(4):2224–2231, 2017.
- [87] G. Loianno, M. Watterson, and V. Kumar. Visual Inertial Odometry for Quadrotors on SE(3). In *Proc. of IEEE International Conference on Robotics and Automation*, 2016.
- [88] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, Nov. 2004.

- [89] B. D. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proc. of the International Joint Conference on Artificial Intelligence*, 1981.
- [90] P. S. Maybeck. Stochastic Models, Estimation and Control. *Academic Press, New York*, 1979.
- [91] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar. The grasp multiple micro-uav testbed. *Robotics Automation Magazine, IEEE*, 17(3):56–65, 2010.
- [92] M. Miller, S. J. Chung, and S. Hutchinson. The VisualInertial Canoe Dataset. *International Journal of Robotics Research*, 2018.
- [93] H.-G. Min, X.-C. Li, P.-P. Sun, X.-M. Zhao, and Z.-G. Xu. Visual Odometry for On-Road Vehicles Based on Trifocal Tensor. In *Proc. of IEEE First International Smart Cities Conference*, 2015.
- [94] J. M. M. Montiel, J. Civera, and A. J. Davison. Unified Inverse Depth Parametrization for Monocular SLAM. *Robotics: Science and Systems*, 2006.
- [95] A. I. Mourikis and S. I. Roumeliotis. A multi-state constraint kalman filter for vision-aided inertial navigation. In *Proc. of IEEE International Conference on Robotics and Automation*, 2007.
- [96] A. I. Mourikis and S. I. Roumeliotis. A dual-layer estimator architecture for long-term localization. In *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2008.
- [97] J. Mu and Y.-L. Cai. Iterated cubature kalman filter and its application. In *Proc. of IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems*, 2011.

- [98] M. W. Mueller, M. Hamer, and R. D. Andrea. Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadcopter state estimation. In *Proc. of IEEE International Conference on Robotics and Automation*, 2015.
- [99] M. W. Mueller, M. Hamer, and R. D. Andrea. Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadcopter state estimation. In *Proc. of IEEE International Conference on Robotics and Automation*, 2015.
- [100] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: A versatile and accurate monocular slam. *IEEE Transactions on Robotics*, 31(5), 2015.
- [101] R. Mur-Artal and J. D. Tardos. ORB-SLAM2 : an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras. *IEEE Transactions on Robotics*, 33(5), 2017.
- [102] R. Mur-Artal and J. D. Tardos. Visual-Inertial Monocular SLAM With Map Reuse. *IEEE Robotics & Automation Letters*, 2(2), 2017.
- [103] A. G. Mutambara. *Decentralized Estimation and Control for Multisensor Systems*. CRC Press, 1998.
- [104] E. D. Nerurkar, K. J. Wu, and S. I. Roumeliotis. C-KLAM: Constrained keyframe-based localization and mapping. *Proc. of IEEE International Conference on Robotics and Automation*, 2014.
- [105] J. Nikolic, J. Rehder, M. Burri, P. Gohl, S. Leutenegger, P. T. Furgale, and R. Siegwart. A Synchronized Visual-Inertial Sensor System with FPGA Pre-Processing for Accurate Real-Time SLAM. In *Proc. of IEEE International Conference on Robotics and Automation*, 2014.
- [106] E. Olson. AprilTag: A robust and flexible visual fiducial system. In *Proc. of IEEE International Conference on Robotics and Automation*, pages 3400–3407. IEEE, May 2011.

- [107] L. L. S. Ong, M. Ridley, J.-h. Kim, E. Nettleton, and S. Sukkarieh. Six DoF Decentralised SLAM. In *Proc. of Australian Conference on Robotics and Automation*, 2006.
- [108] K. Pakki, B. Chandra, D.-W. Gu, and I. Postlethwaite. Cubature Information Filter and its Applications. In *Proc. of American Control Conference*, 2011.
- [109] K. Pakki, B. Chandra, D.-W. Gu, and I. Postlethwaite. Square root cubature information filter. *IEEE Sensors Journal*, 13(2), 2013.
- [110] G. Panahandeh and M. Jansson. Vision-aided inertial navigation based on ground plane feature detection. *IEEE/ASME Transactions on Mechatronics*, 2014.
- [111] D. Pascarella, S. Venticinque, R. Aversa, M. Mattei, and L. Blasi. Parallel and distributed computing for UAVs trajectory planning. *Journal of Ambient Intelligence and Humanized Computing*, 6, 2015.
- [112] V. Peretroukhin, L. Clement, M. Giamou, and J. Kelly. PROBE : Predictive Robust Estimation for Visual-Inertial Navigation. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.
- [113] V. Peretroukhin, L. Clement, and J. Kelly. Reducing Drift in Visual Odometry by Inferring Sun Direction Using a Bayesian Convolutional Neural Network. In *Proc. of IEEE International Conference on Robotics & Automation*, 2017.
- [114] V. Peretroukhin, W. Vega-brown, N. Roy, and J. Kelly. PROBE-GK : Predictive Robust Estimation using Generalized Kernels. In *Proc. of IEEE International Conference on Robotics & Automation*, 2016.
- [115] B. Pfrommer, N. Sanket, K. Daniilidis, and J. Cleveland. PenncoSyvio: A challenging visual inertial odometry benchmark. In *Proc. of IEEE International Conference on Robotics and Automation*.

- [116] T. Pire, T. Fischer, G. Castro, P. De Cristóforis, J. Civera, and J. Jacobo Berllés. S-PTAM: Stereo Parallel Tracking and Mapping. *Robotics and Autonomous Systems (RAS)*, 93, 2017.
- [117] T. E. Potok, C. Schuman, S. Young, R. Patton, F. Spedalieri, J. Liu, K.-T. Yao, G. Rose, and G. Chakma. A Study of Complex Deep Learning Networks on High-Performance, Neuro-morphic, and Quantum Computers. *ACM Journal on Emerging Technologies in Computing Systems*, 2018.
- [118] G. Qian, R. Chellappa, Q. Zheng, and J. Ortolfo. Camera Motion Estimation Using Monocular Image Sequences and Inertial Data. In *Tech. Rep. CS-TR-3997, University of Maryland, College Park, MD*, 1999.
- [119] T. Qin, P. Li, and S. Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.
- [120] K. Qiu and S. Shen. Model-aided monocular visual-inertial state estimation and dense mapping. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.
- [121] F. Rameau, H. Ha, K. Joo, J. Choi, and I. Kweon. A Real-Time Vehicular Vision System to Seamlessly See-Through Cars. In *Proc. of European Conference on Computer Vision*, 2016.
- [122] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart. Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes. *Proc. of IEEE International Conference on Robotics and Automation*, 2016.
- [123] S. I. Roumeliotis, A. E. Johnson, and J. F. Montgomery. Augmenting Inertial Navigation with Image-Based Motion Estimation. In *Proc. of IEEE International Conference on Robotics & Automation*, 2002.

- [124] I. Sa, M. Kamel, M. Burri, M. Bloesch, R. Khanna, M. Popovic, J. Nieto, and R. Siegwart. Build your own visual-inertial drone: A cost-effective and open-source autonomous drone. *IEEE Robotics and Automation Magazine*, 25, 2018.
- [125] F. Santoso, M. A. Garratt, and S. G. Anavatti. Visual-inertial navigation systems for aerial robotics: Sensor fusion and technology. *IEEE Transactions on Automation Science and Engineering*, 14(1), 2017.
- [126] T. Sayre-McCord, W. Guerra, A. Antonini, J. Arneberg, A. Brown, G. Cavaleiro, Y. Fang, A. Gorodetsky, D. McCoy, S. Quilter, F. Riether, E. Tal, Y. Terzioglu, L. Carlone, and S. Karaman. Visual-Inertial Navigation Algorithm Development Using Photorealistic Camera Simulation in the Loop. In *Proc. of IEEE International Conference on Robotics and Automation*, 2018.
- [127] D. Scaramuzza, M. C. Achtelik, L. Doitsidis, F. Friedrich, E. Kosmatopoulos, A. Martinelli, M. W. Achtelik, M. Chli, S. Chatzichristofis, L. Kneip, D. Gurdan, L. Heng, G. H. Lee, S. Lynen, M. Pollefeys, A. Renzaglia, R. Siegwart, J. C. Stumpf, P. Tanskanen, C. Troiani, S. Weiss, and L. Meier. Vision-controlled micro flying robots: From system design to autonomous navigation and mapping in gps-denied environments. *IEEE Robotics Automation Magazine*, 21(3):26–40, Sept 2014.
- [128] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, and R. Siegwart. maplab: An Open Framework for Research in Visual-inertial Mapping and Localization. *IEEE Robotics and Automation Letters*, 3(3), 2018.
- [129] J. Shi and C. Tomasi. Good Features to Track. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

- [130] D. W. Strelow. *Motion estimation from image and inertial measurements*. PhD thesis, Carnegie Mellon University, 2004.
- [131] A. Suleiman, Z. Zhang, L. Carlone, S. Karaman, and V. Sze. Navion : A Fully Integrated Energy-Efficient Visual-Inertial Odometry Accelerator for Autonomous Navigation of Nano Drones. In *Proc. of IEEE Symposium on VLSI Circuits*, 2018.
- [132] K. Sun, K. Mohta, B. Pfrommer, M. Watterson, S. Liu, Y. Mulgaonkar, C. J. Taylor, and V. Kumar. Robust Stereo Visual Inertial Odometry for Fast Autonomous Flight. *IEEE Robotics And Automation Letters*, 3(2), 2018.
- [133] J. J. Tarrio and S. Pedre. Realtime edge-based visual odometry for a monocular camera. In *Proc. of IEEE International Conference on Computer Vision*, 2015.
- [134] L. Teixeira, I. Alzugaray, and M. Chli. Autonomous Aerial Inspection Using Visual-Inertial Robust Localization and Mapping. *Field and Service Robotics*, 2018.
- [135] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography:a factorization method. *International Journal on Computer Vision*, 9(2), 1992.
- [136] M. Trajkovii and M. Hedley. Fast corner detection. *Image and Vision Computing*, 1998.
- [137] N. Trawny and S. I. Roumeliotis. Indirect kalman filter for 3d attitude estimation: A tutorial for quaternion algebra. *Technical Report, University of Minnesota, Department of Computing Science and Engineering*, 2005.
- [138] V. Usenko, J. Engel, and D. Cremers. Direct Visual-Inertial Odometry with Stereo Cameras. In *Proc. of IEEE International Conference on Robotics and Automation*, 2016.
- [139] R. G. Valenti, I. Dryanovski, C. Jaramillo, D. P. Strom, and J. Xiao. Autonomous Quadrotor Flight Using Onboard RGB-D Visual Odometry. In *Proc. of IEEE International Conference on Robotics & Automation*, 2014.

- [140] M. Vo, S. G. Narasimhan, and Y. Sheikh. Spatiotemporal Bundle Adjustment for Dynamic 3D Reconstruction. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [141] L. von Stumberg, V. Usenko, and D. Cremers. Direct Sparse Visual-Inertial Odometry using Dynamic Marginalization. In *Proc. of IEEE International Conference on Robotics and Automation*, 2018.
- [142] E. A. Wan and R. V. D. Merwe. The unscented kalman filter for nonlinear estimation. In *Proc. of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000.
- [143] J. Wan and R. Zhan. Iterated unscented kalman filter for passive target tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 43(3), 2007.
- [144] C. Wang, H. Zhang, T.-m. Nguyen, and L. Xie. Ultra-Wideband Aided Fast Localization and Mapping System. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.
- [145] S. Wang, J. Feng, and C. K. Tse. A class of stable square-root nonlinear information filters. *IEEE Transactions on Automatic Control*, 2014.
- [146] S. E. Webster, R. M. Eustice, H. Singh, and L. L. Whitcomb. Advances in Single-Beacon One-Way-Travel-Time Acoustic Navigation for Underwater Vehicles. *The International Journal of Robotics Research*, 31(08), 2012.
- [147] S. Weiss and R. Siegwart. Real-Time Metric State Estimation for Modular Vision-Inertial Systems. In *Proc. of IEEE International Conference on Robotics and Automation*, 2011.

- [148] K. Wu, T. Zhang, D. Su, S. Huang, and G. Dissanayake. An Invariant-EKF VINS Algorithm for Improving Consistency. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.
- [149] J. Yang, A. Dani, S. J. Chung, and S. Hutchinson. Vision-based Localization and Robot-centric Mapping in Riverine Environments. *Journal of Field Robotics*, 34(3), 2017.
- [150] Z. Yang, F. Gao, and S. Shen. Real-time Monocular Dense Mapping on Aerial Robots Using Visual-Inertial Fusion. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.
- [151] Z. Yang, F. Gao, and S. Shen. Real-Time Monocular Dense Mapping on Aerial Robots Using Visual-Inertial Fusion. In *Proc. of IEEE International Conference on Robotics and Automation*, 2017.
- [152] H. Yu and A. I. Mourikis. Vision-Aided Inertial Navigation with Line Features and a Rolling-Shutter Camera. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.
- [153] H. Yu and A. I. Mourikis. Edge-based Visual-Inertial Odometry. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.
- [154] J. Zhang and S. Singh. Visual-lidar Odometry and Mapping : Low-drift , Robust , and Fast. In *Proc. of IEEE International Conference on Robotics and Automation*, 2015.
- [155] T. Zhang, K. Wu, J. Song, S. Huang, and G. Dissanayake. Convergence and Consistency Analysis for A 3D Invariant-EKF SLAM. *IEEE Robotics and Automation Letters*, 2017.
- [156] Z. Zhang, C. Ma, and R. Zhu. A FPGA-based, granularity-variable neuromorphic processor and its application in a MIMO real-time control system. *Sensors (Switzerland)*, 2017.

- [157] Z. Zhang, A. Suleiman, L. Carlone, V. Sze, and S. Karaman. Visual-Inertial Odometry on Chip : An Algorithm-and-Hardware Co-design Approach. *Robotics: Science and Systems*, 2017.
- [158] S. Zhao, F. Lin, K. Peng, X. Dong, B. M. Chen, and T. H. Lee. Vision-aided Estimation of Attitude, Velocity, and Inertial Measurement Bias for UAV Stabilization. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 2016.
- [159] Y. Zhao and P. A. Vela. Good Feature Selection for Least Squares Pose Optimization in VO / VSLAM. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018.
- [160] X. Zheng, Z. Moratto, M. Li, and A. I. Mourikis. Photometric Patch-based Visual-Inertial Odometry. In *Proc. of IEEE International Conference on Robotics and Automation*, 2017.